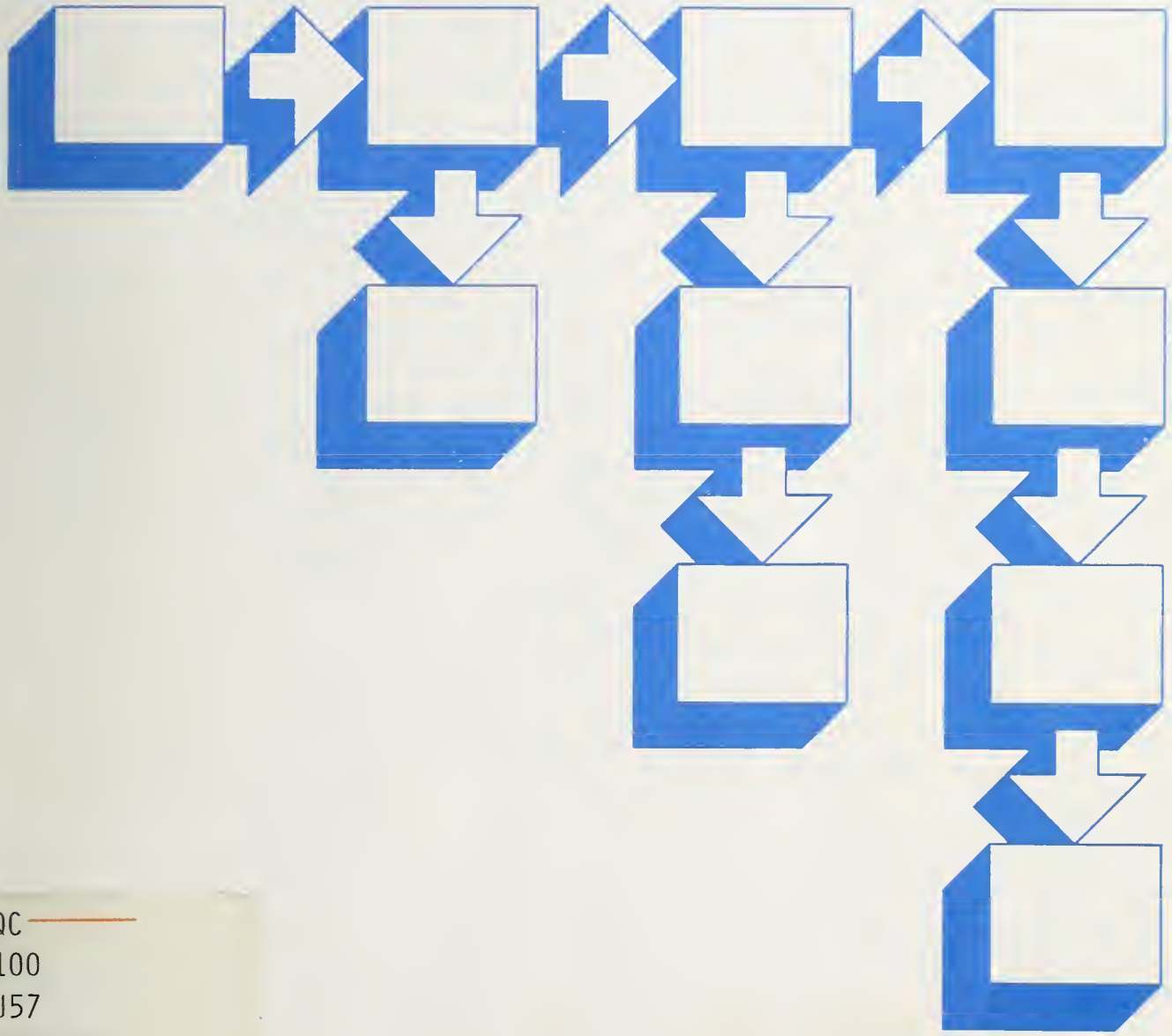
Computer Science and Technology

NBS Special Publication 500-151

A Knowledge-Based System for Physical Database Design

Christopher E. Dabrowski and David K. Jefferson



The National Bureau of Standards[1] was established by an act of Congress on March 3, 1901. The Bureau's overall goal is to strengthen and advance the nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research to assure international competitiveness and leadership of U.S. industry, science and technology. NBS work involves development and transfer of measurements, standards and related science and technology, in support of continually improving U.S. productivity, product quality and reliability, innovation and underlying science and engineering. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, the Institute for Computer Sciences and Technology, and the Institute for Materials Science and Engineering.

The National Measurement Laboratory

Provides the national system of physical and chemical measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; provides advisory and research services to other Government agencies; conducts physical and chemical research; develops, produces, and distributes Standard Reference Materials; provides calibration services; and manages the National Standard Reference Data System. The Laboratory consists of the following centers:

- Basic Standards[2]
- Radiation Research
- Chemical Physics
- Analytical Chemistry

The National Engineering Laboratory

Provides technology and technical services to the public and private sectors to address national needs and to solve national problems; conducts research in engineering and applied science in support of these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the ultimate user. The Laboratory consists of the following centers:

- Applied Mathematics
- Electronics and Electrical Engineering[2]
- Manufacturing Engineering
- Building Technology
- Fire Research
- Chemical Engineering[3]

The Institute for Computer Sciences and Technology

Conducts research and provides scientific and technical services to aid Federal agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following divisions:

- Information Systems Engineering
- Systems and Software Technology
- Computer Security
- System and Network Architecture
- Advanced Systems

The Institute for Materials Science and Engineering

Conducts research and provides measurements, data, standards, reference materials, quantitative understanding and other technical information fundamental to the processing, structure, properties and performance of materials; addresses the scientific basis for new advanced materials technologies; plans research around cross-cutting scientific themes such as nondestructive evaluation and phase diagram development; oversees Bureau-wide technical programs in nuclear reactor radiation research and nondestructive evaluation; and broadly disseminates generic technical information resulting from its programs. The Institute consists of the following Divisions:

- Ceramics
- Fracture and Deformation[3]
- Polymers
- Metallurgy
- Reactor Radiation

[1]Headquarters and Laboratories at Gaithersburg, MD, unless otherwise noted; mailing address Gaithersburg, MD 20899.
[2]Some divisions within the center are located at Boulder, CO 80303.
[3]Located at Boulder, CO, with some elements at Gaithersburg, MD

Computer Science and Technology

NBS Special Publication 500-151

A Knowledge-Based System for Physical Database Design

Christopher E. Dabrowski and David K. Jefferson

Information Systems Engineering Division
Institute for Computer Sciences and Technology
National Bureau of Standards
Gaithersburg, Maryland 20899

February 1988

U.S. DEPARTMENT OF COMMERCE
C. William Verity, Secretary

National Bureau of Standards
Ernest Ambler, Director

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

Library of Congress Catalog Card Number: 88-600502
National Bureau of Standards Special Publication 500-151
Natl. Bur. Stand. (U.S.), Spec. Publ. 500-151, 59 pages (Feb. 1988)
CODEN: XNBSAV

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1988

For sale by the Superintendent of Documents, U.S. Government Printing Office, Washington DC 20402

A KNOWLEDGE-BASED SYSTEM FOR PHYSICAL DATABASE DESIGN

by

Christopher E. Dabrowski
David K. Jefferson

Information Systems Engineering Division
Institute for Computer Sciences and Technology
National Bureau of Standards

ABSTRACT

A knowledge-based system for physical database design has been developed at the Institute for Computer Sciences and Technology. This system processes large multi-entity logical data structures with complex workload requirements and identifies near-optimal physical designs. It employs heuristics developed by physical design experts and cost modeling algorithms to reduce the large number of design alternatives available in large complex problems to a few select designs. This system is implemented in Lisp.

Key words: certainty factor; entity-relationship model; inference engine; knowledge engineering; knowledge-based system; logical data structure; logical database design; physical database design; rule-based system.

PREFACE

Database technology has progressed rapidly over the last two decades. Enterprise-wide sharing of data resources is now common practice. A body of knowledge has evolved to aid in the design of efficient and functionally sufficient databases. Database design requires information about how the data will be used, the characteristics of the supporting computer hardware and database management system, and an experienced practitioner to perform database design.

The task of professionals working on physical database design has become much more complicated. As information systems have expanded in recent years, the size and complexity of database design problems have mushroomed. Databases which were originally developed to support a particular application, but later became data resources for other areas, are now giving way to truly integrated data resources. Integration will be the focus of the next decade.

Experience has led to the realization that the problem of arriving at an optimal physical database design for a large, complex system may not be humanly possible without the aid of software tools. This report describes the prototype implementation of a software tool that uses heuristics for physical database design. This is a tool that can, for the first time, make near-optimal physical designs feasible for large databases. A framework has been laid for further research into knowledge-based systems that solve database design problems by incorporating more information about data structure and data use.

I hope this report will stimulate the recognition that much more work is needed in this area, and that software tools are essential to solve the problems of large database design.

Mary Mitchell
Integrated Systems Group
Factory Automation Systems Division
National Bureau of Standards
February, 1988

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

In this report we describe our progress at the National Bureau of Standards in building a knowledge-based system (KBS) for physical database design. This is a new approach to the solution of very difficult physical database design problems. The report is intended for an audience of database specialists, and therefore assumes that the reader is familiar with concepts such as the Entity-Relationship Model and Database Language SQL.

Logical database design is the process of determining an information system structure which is independent of software or hardware considerations. The objectives of our work in logical database design are to improve the effectiveness of an information system by maximizing the consistency and completeness of the database [FONG85].

Physical database design is the implementation of a logical design in a particular computer system environment. The objectives of our work in physical database design are to improve the performance of the information system by minimizing retrieval time, update time, and storage cost.

For large, logically complex databases, physical design is an extremely difficult task. Typically, an enormous number of alternatives must be explored in searching for a good physical design. Often, optimal or near-optimal designs cannot be discovered, resulting in the creation of inefficient and costly databases.

A human designer can formulate rich sets of largely qualitative rules to focus on critical parts of a design problem, to recognize special cases, and to reduce the number of reasonable alternative designs. However, the application of these rules may be difficult, due to human limitations in performing large numbers of such simple data processing chores as estimating retrieval times and keeping track of the different alternatives. Human designers have difficulty performing the detailed evaluation and comparison of even a modest number of alternatives; consequently, the human designer will tend to drop many promising alternatives.

In contrast, conventional algorithmic software systems have relatively small sets of qualitative rules [CARL80]. The rule set for a conventional system may greatly restrict the possible alternative solutions that are considered, in which case the set of alternatives may be much too small to include good solutions, or it may generate too many alternatives, in which case it will be impossible to evaluate all of them. However, if a good set of alternatives is available, the conventional system can handle the quantitative evaluation of many more alternatives than the human designer.

The objective of our knowledge-based system is to combine the best features of the human and algorithmic designers. The system uses a knowledge base derived from human experience to generate a reasonably small set of alternatives with a reasonably large chance of containing a good solution. The general approach is to divide the large problem into many smaller problems that are then independently and efficiently fine-tuned by a conventional algorithmic system [MARC78]. The knowledge-based system performs much the same task as the database administrator in [CARL80].

2. OVERVIEW OF PHYSICAL DATABASE DESIGN

This section gives a synopsis of physical database design and discusses some of the problems and difficulties encountered by human designers and by software design systems based on conventional programming techniques.

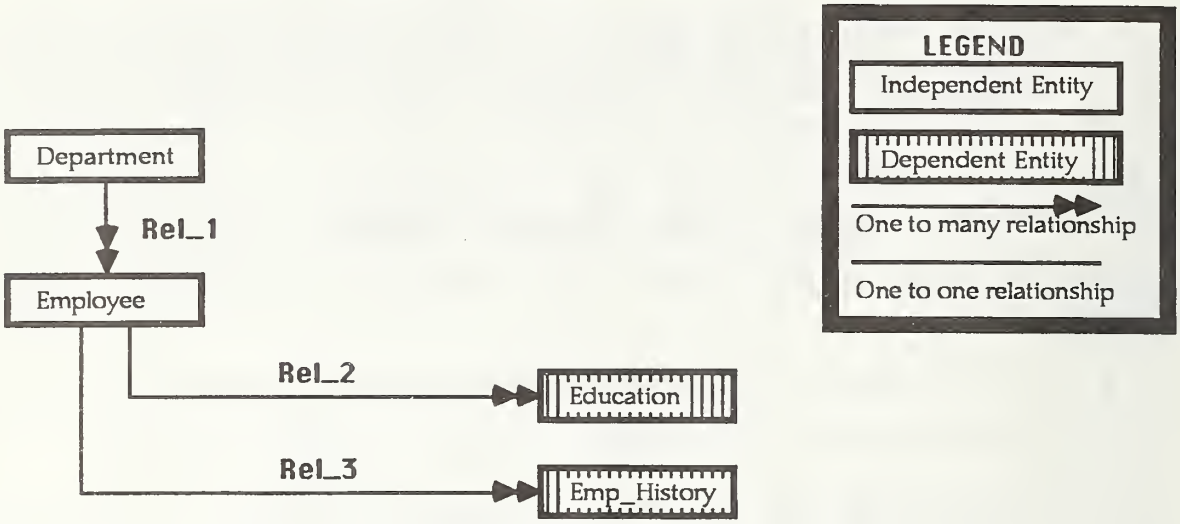
2.1 INFORMATION REQUIRED BY PHYSICAL DATABASE DESIGN

Five categories of information are required by physical database design:

- the structural characteristics of the information system,
- the retrieval and update workload for the system,
- the parameters for the hardware environment in which the physical design will be implemented,
- the physical structures which are provided by the software environment, and
- a means for determining a single cost for a physical design.

2.1.1 Logical Data Structure

We use a very simple entity-relationship-attribute model to represent the structural characteristics of an information system, and to provide a basis for describing a physical database design. The logical design phase produces a Logical Data Structure (LDS) consisting of a number of entities connected by one to one (1 to 1) or one to many (1 to M) relationships, both subject to appropriate integrity constraints. Figure 1 is an example of an LDS diagram with entities (boxes) and relationships (lines); attributes have been omitted because they are not explicitly referenced in this paper.



This diagram displays three relationships which connect four entities. One to many (1 to M) relationships are represented by double arrows drawn near the "M" entity. For example, there may be many instances of EMPLOYEEs for an instance of a DEPARTMENT, but there will be only one instance of a DEPARTMENT for any given instance of EMPLOYEE. A relationship may be described in two directions with each direction having a unique descriptor name derived from the primary identifiers of the entities. For example, the descriptor names for REL_1 are EMPLOYEES_OF_DEPARTMENT and DEPARTMENT_OF_EMPLOYEES.

Entities in boxes with vertical stripes are dependent entities. For the purposes of this paper, a dependent entity will be an "M" entity which is dependent on a "1" entity in a one to many relationship (i.e., the mapping is onto). An entity that is not dependent on any other entity is an independent entity. Independent entities are in unstriped boxes. In Figure 1, DEPARTMENT and EMPLOYEE are independent entities, while EDUCATION and EMP_HISTORY are dependent entities.

Figure 1. A Small Portion of an LDS Diagram

2.1.2 Workload

The set of database retrievals and updates, or workload, is a very important and often complex aspect of an information system. End-user queries may be obtained to describe retrieval usage. Queries may be quite complex, traversing many relationships and involving a sequence of entities.

Figure 2 represents the retrieval workload for the small LDS portion shown in Figure 1. Four retrievals are shown. Database Language SQL [ANSI86], plus quantitative parameters, has been used in this example, because SQL is a commonly used, standard database language[1]. A complex retrieval is composed of a number of contexts, each of which deals with one entity (but possibly retrieving many instances of that entity). Individual contexts are described by selection criteria, projection criteria, and ordering criteria. Each context has an associated frequency. Frequency is the number of times the context is executed per month. Each context also specifies the average proportion of record instances retrieved during each execution. Retrieval activity is forwarded from one context to the next; that is, the activity continues at the latter entity in the context of the former. For example, {Retrieval 1} of Figure 2 has two contexts; the first dealing with EMPLOYEE and the second with EDUCATION. The EMPLOYEE context is executed 500 times per month with 25% of the instances retrieved during each execution. Activity is forwarded to the EDUCATION context which is executed 12500 times per month with 25% of the instances retrieved during each execution. Context may be very important for performance. In {Retrieval 1}, the retrieval of a large number of EDUCATION records in the context of a specific EMPLOYEE can be much less costly than the retrieval of a similar number of random EDUCATION records. A large subset retrieval is the retrieval of a sufficiently large proportion of records to require a sequential scan for efficient search.

Update workload is also important. It is described by the frequency of insertion and deletion of instances of entities and the frequency of modification of entity attributes. We use the term workload complexity to indicate a measure of the number of different retrievals, the variability of their size and frequency, and the amount of update workload. The importance and complexity of physical database design increases rapidly as workload complexity increases.

---

[1]The actual language used in our system is navigational [CARL80]. A future enhancement would be a query optimization phase that would translate SQL into a navigational form. The reader can assume that navigation follows the order in which entities are listed in the SQL. For example, {Retrieval 1} starts at EMPLOYEE and navigates to EDUCATION.

```
{Retrieval 1}
SELECT  EMPLOYEE_NAME, SSN, INSTITUTION_NAME, MAJOR
    FROM    EMPLOYEE,   -- {FREQUENCY 500, PROPORTION 0.25}
        EDUCATION   -- {FREQUENCY 125000, PROPORTION 0.25}
    WHERE   EDUCATION.EMPID = SSN AND AGE < 30


{Retrieval 2}
SELECT  *
    FROM    EMPLOYEE    -- {FREQUENCY 1000, PROPORTION 1.0}


{Retrieval 3}
SELECT  DEPARTMENT_NAME, EMPLOYEE_NAME, SSN, AGE
    FROM    DEPARTMENT, -- {FREQUENCY 500, PROPORTION 0.5}
        EMPLOYEE    {FREQUENCY 300000, PROPORTION 0.3}
    WHERE   EMPLOYEE.DEPT = DEPARTMENT_NAME AND AGE >50


{Retrieval 4}
SELECT  EMPLOYEE_NAME, SSN, EMPLOYER, START_DATE, END_DATE
    FROM    EMPLOYEE    -- {FREQUENCY 10, PROPORTION 0.001}
        EMP_HISTORY {FREQUENCY 10, PROPORTION 0.001 }
    WHERE EMPLOYEE.SSN = ?
```

All retrievals have multiple contexts except {Retrieval 2}. {Retrieval 4} contains a small subset retrieval of EMPLOYEE. {Retrieval 1} is a large subset retrieval with two contexts. The first is a retrieval on EMPLOYEE, selecting individuals under 30 years of age. This context is executed 500 times per month, and 0.25 of the EMPLOYEE records are retrieved in each execution. The second context is on EDUCATION, retrieving those instances belonging to qualifying EMPLOYEE instances. This context is retrieved with a frequency of 125000 times per month, with 0.25 of the EDUCATION records retrieved in each execution.

Figure 2. Retrievals for Figure 1.

2.1.3 Hardware Environment

The third component is the hardware environment in which the physical database will exist. The following are the major parameters needed to characterize a hardware environment:

- Average time required for random access to a track.
- Average time required for sequential access to a track.
- Length of a track.

2.1.4 Software Environment

The fourth component is the software environment, which will impose restrictions on the possible physical structures. The currently modeled software environments are:

- A generalized model based on frame memory [MARC78, CARL80] which includes variable length records, record clustering, record segmentation, hierarchical index and sequential scan for primary access, and inverted files (block granularity) and sequential scan for secondary access. Files may be ordered to reduce the costs of sorting.
- The CODASYL model [DDLC78], including record clustering, and owner-member sets implemented by either ring lists (including back pointers and owner pointers) or pointer arrays.

2.1.5 Cost

The fifth and final category of information is a means for determining the quality of a partial or complete design. This requires a way of combining into one cost the major cost components: retrieval time, update time, and the size of the required storage. The current cost estimate of the knowledge-based system is simply the unweighted sum of the times required to perform updates and retrievals. The fine-tuning performed by the conventional system is a weighted sum of those times and the storage costs.

2.2 GENERAL CHARACTERISTICS OF PHYSICAL DATABASE DESIGN

Physical database design is the process of determining physical structures for the entities and relationships such that the overall cost is reasonable. The objective should be to achieve an optimal or near-optimal design, but for most real problems

there is no way of knowing whether or not this objective has been achieved.

The first step in physical database design is generally to temporarily simplify the problem: to ignore much of the workload complexity, and to focus on the more critical aspects (e.g., the most frequent retrievals).

The next and primary step in physical database design is the selection of relationship representations (i.e., the physical level implementations of the relationships specified in the LDS). Selection of representations provides the basis for the formation of physical records and for the physical relationships among those records. Representations are of three generic types:

- o absorption, where the two entities of a relationship are stored in the same physical area;
- o symbolic pointer, where one entity contains the logical identifier of the other; and
- o direct pointer, where one entity contains the physical address of the other.

These representations may be used in combination; e.g., both a direct and a symbolic pointer could be used to represent the relationship from a dependent entity to an entity on which it is dependent. Carlis [CARL80] has identified 10 possible combinations of the generic representations for relationships involving dependent entities and 17 possible combinations for relationships where no dependency between entities exists.[2]

Following the selection of representations, which produces the outline of a database design, fine-tuning is required to determine record segmentations, indexes and other detailed primary and secondary access methods, block sizes, secondary memory management for overflow handling, and initial loading factors. This is a critical step to improving performance, and is well-suited to an algorithmic solution by conventional programming methods [MARC78, CARL80].

---

[2]For an LDS of moderate size, say 100 relationships, the number of possible combinations of relationship representations is at least 10**100. Even if heuristics are used to reduce the 10 or 17 to 3 reasonable representations, the number of possible alternatives is 3**100, which is computationally infeasible. Clearly, physical database design cannot be approached by a brute force enumeration of possible designs.

2.3 HOW PHYSICAL DATABASE DESIGN COULD BE DONE BY HUMANS

The following is an outline of a reasonable methodology for human designers:

- Select each relationship with a low workload complexity and assign a representation based on the structural and workload characteristics of that representation alone (i.e., ignore interactions with other representation decisions).
- For each of the remaining relationships, select two or three candidate representations for each relationship, using both structural and workload characteristics. For especially problematic areas, many alternatives may be considered.
- Evaluate combinations of candidate representations.
- Fine-tune the design as above.

To illustrate the complexity of the design problem, we will use the fragment of Figure 1 which is shown in Figure 3.

Figure 3. Selection of a Relationship Representation

Let us assume that the primary identifier of EDUCATION is a compound key containing the primary identifier of EMPLOYEE. EDUCATION is dependent on EMPLOYEE. The human designer should conclude from structural rules that EDUCATION could be absorbed into EMPLOYEE as a repeating group, resulting in a relatively large EMPLOYEE file.

After examining workload requirements, the designer could conclude that {Retrieval 2} of Figure 2 represents a significant amount of large subset retrieval on EMPLOYEE. We refer to such a conclusion as a characterization of the retrieval activity on the EMPLOYEE. The large subset activity **is not** forwarded along relationship REL_2 to the EDUCATION entity. In other words, the retrieval activity directed to EMPLOYEE is not followed by subsequent activity directed to EDUCATION. Based on this information, the human expert might characterize EMPLOYEE as having a substantial amount of large subset retrieval activity

and characterize REL_2 as a relationship which has light activity and little workload association between its entities. This may lead the designer to represent relationship REL_2 with a symbolic pointer from EDUCATION to EMPLOYEE. The entities would be stored separately, resulting in a smaller EMPLOYEE file, which would reduce the cost of the large subset retrieval.

{Retrieval 1} of Figure 2 represents a smaller but substantial amount of large subset activity on EMPLOYEE which is forwarded along REL_2. This information might lead the designer to modify the original characterization of the workload on REL_2 to include significant activity on REL_2. This, together with the structural dependence of EDUCATION on EMPLOYEE, would argue for the absorption of EDUCATION into EMPLOYEE.

After considering all the facts, the designer would most probably want to try both representations, and compare the results after designing detailed file organizations. Actual problems involve many more considerations and result in enormous numbers of combinations of alternatives.

After selection of relationship representations, the designer may divide the LDS into clusters. LDS clusters are smaller subdivisions of the LDS. Ideally, structural and workload interrelationships are strong within clusters and weak among clusters, so that each cluster can be designed separately without significant degradation of performance of the composite design. Within each cluster, alternative relationship representations are varied in an attempt to determine the lowest cost combinations. Each unique combination of alternatives is known as a skeleton.

Non-cyclic physical groupings of one or more entities connected by absorption form structures which we will refer to as canonical records. Canonical records are prototype physical records which may be segmented during the detailed fine-tuning of the physical database design. In the example given above we discussed three possible canonical records: one for EMPLOYEE, another for EDUCATION, and a combined record consisting of an EMPLOYEE and associated EDUCATIONs (a subordinate repeating group).

The example given above is representative of just a small part of physical database design. The process of varying representations and forming records may result in a large number of alternative canonical records, only a few of which are efficient. Further work on individual canonical records consists of fine-tuning as described earlier, which is quite impractical for humans alone.

3. THE KBS APPROACH TO PHYSICAL DATABASE DESIGN

This section provides a brief description of elements of knowledge-based systems technology and a discussion of specific benefits of the internal representation and application of expert knowledge for automated physical database design.

KBS technology provides a way in which the expertise of human **designers can be stored and applied by** a **computer program.** Knowledge about physical database design can be obtained from human experts and conveniently translated into rule form. Rules can be thought of as chunks of knowledge about how to do physical database design. They can be represented in an internal knowledge base and applied by an inference engine to the physical design problem. The problem is itself represented internally as a large database of facts about which the rules can reason. Application of rules results in examination of part or all of the design problem, and the conclusion of new information leading to decisions about which design action to undertake next. The result is a systematic advancement of physical database design for the entire problem.

3.1 RULE-BASED SYSTEMS

Knowledge-based systems, which rely primarily on rules for representing and applying knowledge, have several important aspects which are described in this section. Readers familiar with rule-based systems may omit this section.

Rules consist of IF --> THEN condition action pairs. Rules are internal data structures used to represent small pieces of knowledge about what action to take or what to conclude under a particular set of conditions. Rules have two parts: the IF part, or antecedent, lists one or more conditions which must hold true; the THEN part, or consequent, contains conclusions which are reached if the conditions in the IF part are satisfied. Individual conditions and conclusions are represented internally as clauses or expressions which are patterns to be matched against actual data. Conditions may refer to any data in the knowledge-based system. Facts in the internal database have formats consistent with the patterns in rule clauses. The physical design problem includes facts relating to the structure or workload associated with the LDS, and facts relating to the state of the current design. Conditions may also refer to conclusions reached by other rules. Conclusions in a THEN part may include a characterization of part of the problem, a selection of design structures, or a specification of actions to be taken. A simple example is shown below (variables are represented by names beginning with a question mark):

```
{SAMPLE RULE}
    IF  DEGREE ?Rel ?Ent1 to ?Ent2 1 to M
        CHARACTERIZATION ?Rel ?Ent1 ?Ent2 HIGH-ACTIVITY
    THEN
        POSSIBLE REPRESENTATION: ?Ent1 ABSORBS ?Ent2
```

This rule has two conditions in the IF part and a single conclusion in the THEN part. The rule states that if a 1 to M relationship exists, and the relationship is characterized as having high retrieval activity in the 1 to M direction, then conclude a possible representation: the "1" entity absorbs the "M" entity.

Using the structural information from Figure 1 we know that EMPLOYEE is in a 1 to M relationship with EDUCATION along relationship REL_2. We also know from {Retrieval 1} in Figure 2 that there is a high retrieval frequency (125000) from EMPLOYEE to EDUCATION along REL_2 resulting in the characterization of REL_2 as a high activity relationship. Using this information, if we substitute REL_2 for the variable ?Rel, EMPLOYEE for ?Ent1, and EDUCATION for ?Ent2, we have:

```
{SAMPLE RULE}
    IF  DEGREE REL_2 EMPLOYEE to EDUCATION 1 to M
        CHARACTERIZATION REL_2 EMPLOYEE EDUCATION HIGH-
        ACTIVITY
    THEN
        POSSIBLE REPRESENTATION: EMPLOYEE ABSORBS EDUCATION
```

The resulting conclusion is a possible representation for REL_2: EMPLOYEE absorbs EDUCATION.

Rules may be organized into rule sets by topic or category. They may be segregated into groups on the basis of subject matter, types of conclusions reached, and problems addressed, among other criteria. These groups may be applied to the problem individually at specific times. A determination must be made about when to apply a particular rule set and the conditions under which it may operate. This determination may be made by a controlling module responsible for overall problem processing. Typically, this module is itself a set of rules.

Not all knowledge can be conveniently expressed in rules. Rules may invoke algorithms to compute some facts. Rules may also invoke step-by-step procedures which perform computations.

Rules are applied by an inference engine. Inference engines are computer programs which match the patterns in rules against existing information to make conclusions. They are

responsible for applying the rules of a knowledge base or a subdivision of the knowledge base, and for controlling the execution or "reasoning" of a knowledge-based system. Two strategies are generally recognized:

- Backward chaining begins with a top level goal: to prove that a premise is implied by existing facts. Backward chaining does this by working "backwards" through a series of (hopefully) simpler subgoals which will establish the premise. The procedure is simple: if a required fact is not already known, a rule is sought which includes that fact in the THEN part. The conditions in the IF part of the rule must then be satisfied; each condition becomes a new subgoal. The procedure continues until either the top level goal is established, or no new subgoals can be generated.

- Forward chaining is generally used to determine the consequences of facts. That is, the IF portions of rules are examined to see whether or not they are true. If they are, the facts in the THEN part are concluded and added to the knowledge base. The IF portions are then examined again to see if new facts can be concluded. The process continues until no more new facts can be concluded.

Our knowledge-based system is predominantly backward chaining. It uses individual rule sets for specific functions including a rule set for overall control of the KBS, as we will describe below.

Certainty factors allow for inexact judgment and enhance the reasoning power of the system. Certainty factors are a numeric measure of the degree to which a fact is believed to be true (or false) by the knowledge-based system. Absolute certainty is 1.0; absolute denial is -1.0. Certainties are used to order alternatives; for example, the relationship representations with the higher certainties would be retained for detailed analysis. If a consequent is established by a given rule, then the certainty of that consequent may be provided by a certainty factor associated with that rule, or may be derived from the certainty factors of the facts which satisfied the antecedent portion of that rule. The inference engine is responsible for the derivation of certainty factors. Experts determine certainty factors associated with the rules. In addition, facts which are concluded by applications of several rules may be assigned a combined certainty. A number of algorithms exist for determination of combined certainty [THOM85]. We base our method on the Bernoulli formula [SHAF76]. Using this formula, if two certainty factors C_1 and C_2 are both positive or both

negative and are associated with different rules concluding the same fact, then they are combined using the function $C_3 = C_1 + C_2 * (1 - C_1)$. The final positive and negative factors are combined by simple summation. Certainty factors are useful in making judgmental conclusions, taking into account different and possible conflicting evidence. Use of certainty factors allows the KBS to select a single design alternative from among several possible alternatives and to make "best guess" approximations of the best choice. The following is the previous example with a certainty factor:

```
{SAMPLE RULE}
    IF  DEGREE ?Rel ?Ent1 to ?Ent2 1 to M
        CHARACTERIZATION ?Rel ?Ent1 ?Ent2 HIGH-ACTIVITY
    THEN
        POSSIBLE REPRESENTATION: ?Ent1 ABSORBS ?Ent2
        CERTAINTY FACTOR 0.25
```

This rule now states that if a 1 to M relationship exists, and it is characterized as having high retrieval activity in the 1 to M direction, then conclude, with certainty 0.25, that the "1" entity should absorb the "M" entity.

3.2 ADVANTAGES OF USING RULES TO REPRESENT KNOWLEDGE

Representing knowledge about physical database design in a knowledge-based system provides benefits not available in conventional programming techniques.

Rules allow knowledge to be stated clearly and concisely. Rules permit incorporation of more complex knowledge about problem identification and design than would be possible in a conventional algorithmic program. In rules, knowledge about different aspects of analysis and design may be combined into one piece and applied to a problem. For instance, knowledge about structural aspects and workload requirements may be combined into one rule about selection of representations.

Using rules allows knowledge to be modified easily. The modularity of rules facilitates experimentation with different rules, different approaches, etc., which enhances knowledge about the physical database design process.

Rules allow the system to explain why conclusions were made. This greatly increases the human designer's understanding of the design, increases confidence in the design system, and facilitates experimentation and change.

4. DESCRIPTION OF THE KBS FOR PHYSICAL DATABASE DESIGN

This section provides an overview of the KBS we are developing. It identifies the major components and the functions they perform and discusses the interconnections between them.

The KBS is intended for large, complex LDSs, having a high degree of workload complexity. The KBS is intended for problems where the number of design alternatives is large enough to cause substantial difficulty for both human designers and conventional software design systems. The goal is to identify a small number of efficient canonical records from the many design alternatives.

The KBS solves such problems by applying heuristic design knowledge to avert the combinatorial explosion of design alternatives, reducing the number of possible canonical records to a select set of good records. The primary method of reducing problem size is to divide the LDS into parts. Record formation strategies are then applied to each part, and efficient records are identified. Each record may then be fine-tuned by a conventional algorithmic design system [MARC78, CARL80]. Preliminary results obtained by processing a limited number of LDS problems indicate that the best canonical records can be reliably selected using the approach we will describe.

The current KBS architecture is divided into the High Level Control Module and the Low Level. The High Level Control Module controls the problem solving process. The Low Level consists of knowledge bases and Lisp function modules which perform tasks specified by the High Level Control Module. We will refer to the High Level Control Module simply as the High Level.

Both the High Level and the Low Level knowledge bases are rule sets which operate by backward chaining. The system is implemented in Lisp.

4.1 THE HIGH LEVEL OF THE KBS

The High Level is responsible for overall control of the processing of physical database design. The High Level makes decisions about what type of design related activity is to be performed on a particular problem and determines where within the LDS this activity should take place. We call the results of these decisions High Level design actions.

The High Level may choose from several design actions. Initial characterization of entities and relationships is the conclusion of structural and workload facts about all entities and relationships in a given problem. For example, an entity might be characterized as DEPENDENT, and a relationship might be characterized as having HIGH-ACTIVITY. Initial characterization

and the initial selection of reasonable relationship representations are among the first design actions initiated by the High Level. These actions provide information for later use by other knowledge bases. The initial selection of reasonable representations identifies all the relationship representations which may be considered in later design actions. Both actions are applied to the entire LDS when beginning work on a problem.

The High Level may choose design actions to reduce problem size. This includes dividing the LDS or a large section of the LDS into smaller clusters (a very small LDS will be treated as one cluster and not divided). Problem size may also be reduced by limiting the number of representations for some of the relationships in a cluster. This action reduces the number of alternative skeletons which will need to be examined.

Once clusters have been created, the High Level must select which cluster is to be worked on, and then determine the best way to find efficient canonical records in the chosen cluster. Two possible design actions may be taken to identify the best records. For clusters with few possible skeletons, all the alternative skeletons may be enumerated, and the best records identified and retained. For clusters with many potential skeletons, heuristics are used to create a small number of good skeletons through selective generation of skeletons. These skeletons will contain efficient records.

The High Level reviews and evaluates the results of dividing an LDS. To do this, the High Level recombines adjacent clusters to create a temporary cluster which incorporates some of the entities and relationships from the adjacent clusters. The High Level may then select design actions for record formation in this new cluster. This permits the identification of new potentially efficient canonical records which could not be found by processing the adjacent clusters individually.

The High Level may be summarized as a process which is continually monitoring the overall design state of a problem and recommending design actions to be carried out. The cumulative effect of this process is to take a large LDS, reduce the problem through division into clusters and possibly through reduction of the number of representations within clusters, identify likely efficient records within each cluster, and, where necessary, form temporary clusters to find new records. This process results in a list of potentially useful records which can be fine-tuned.

The actions of the High Level are summarized in Figure 4. A detailed description of each design action will be provided in sections 4.2, 4.3, and 4.4. Section 4.5 describes knowledge bases for implementation of design actions. Figures 5 through 9 illustrate the cumulative effect of the design actions. Figure 5 shows an LDS. Figure 6 shows the results of dividing this LDS

into three clusters. Figures 7 and 8 show the formation of canonical records for each of the three clusters. These canonical records will then undergo fine-tuning by a conventional algorithmic design system. The upper portion of Figure 9 shows the formation of a temporary cluster from parts of Clusters 1 and 3. The lower portion of Figure 9 shows a new canonical record which would not be found in Clusters 1 or 3.

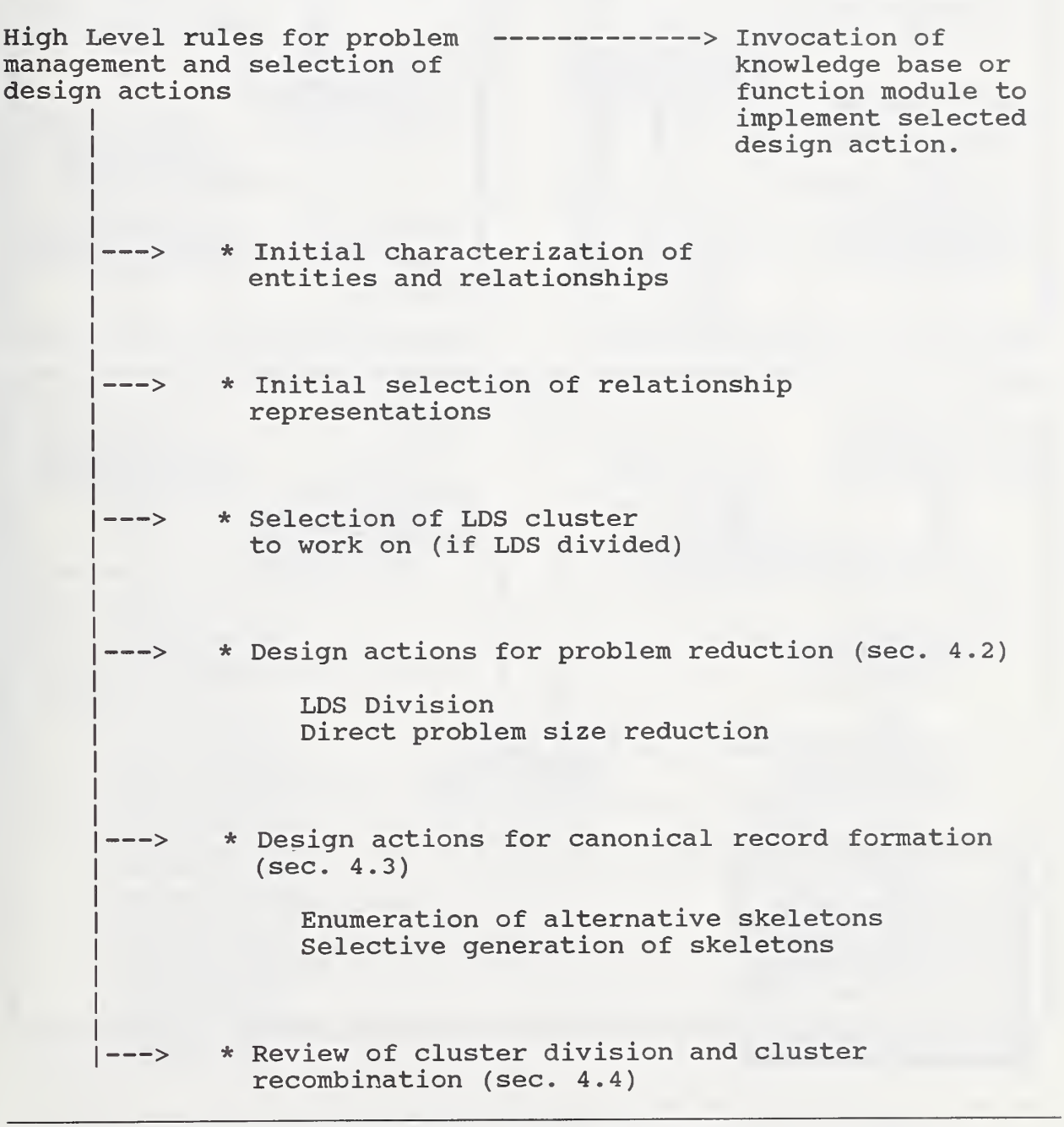


Figure 4. A Diagram of High Level Design Actions

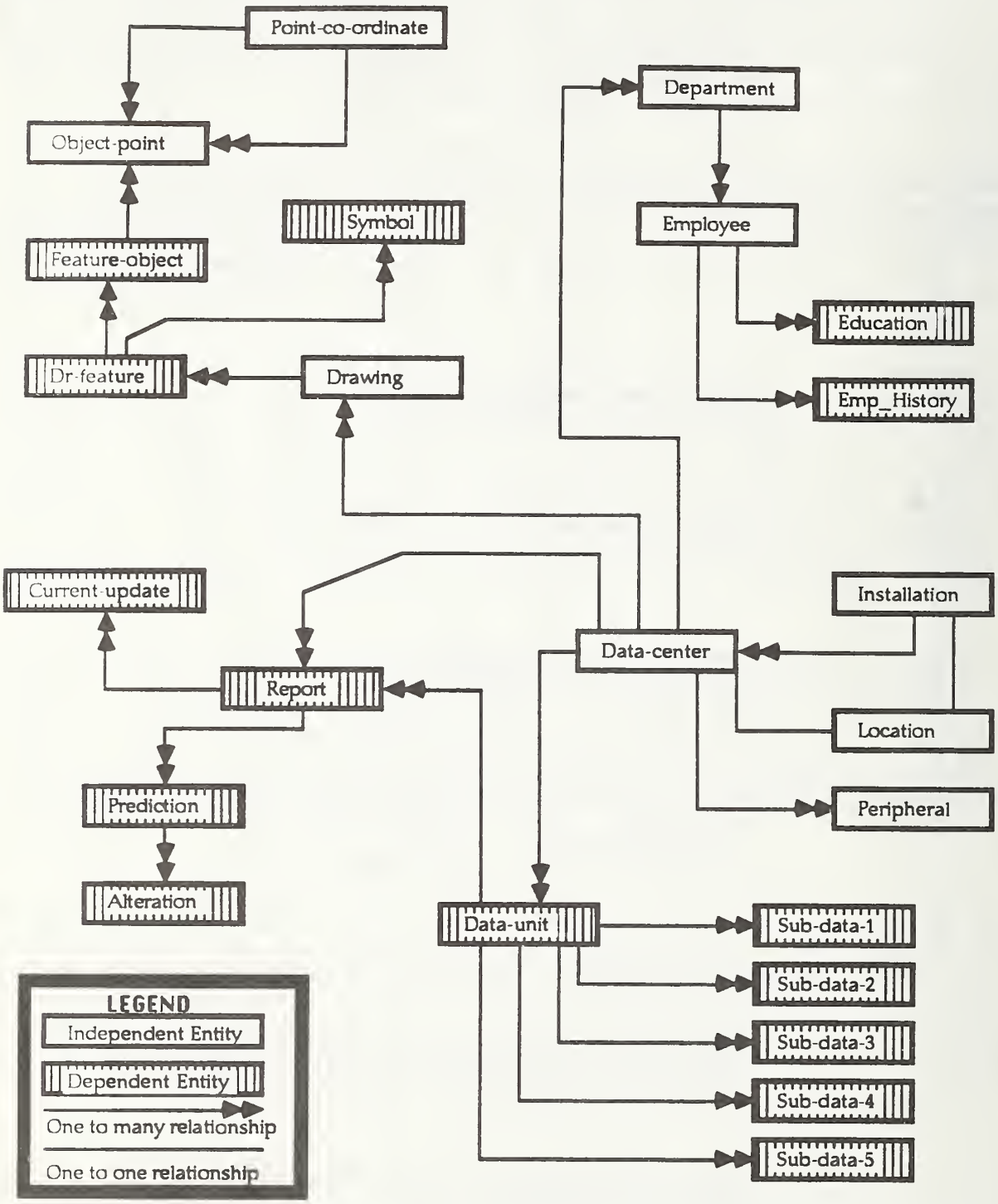


Figure 5. A Sample Logical Data Structure (LDS)

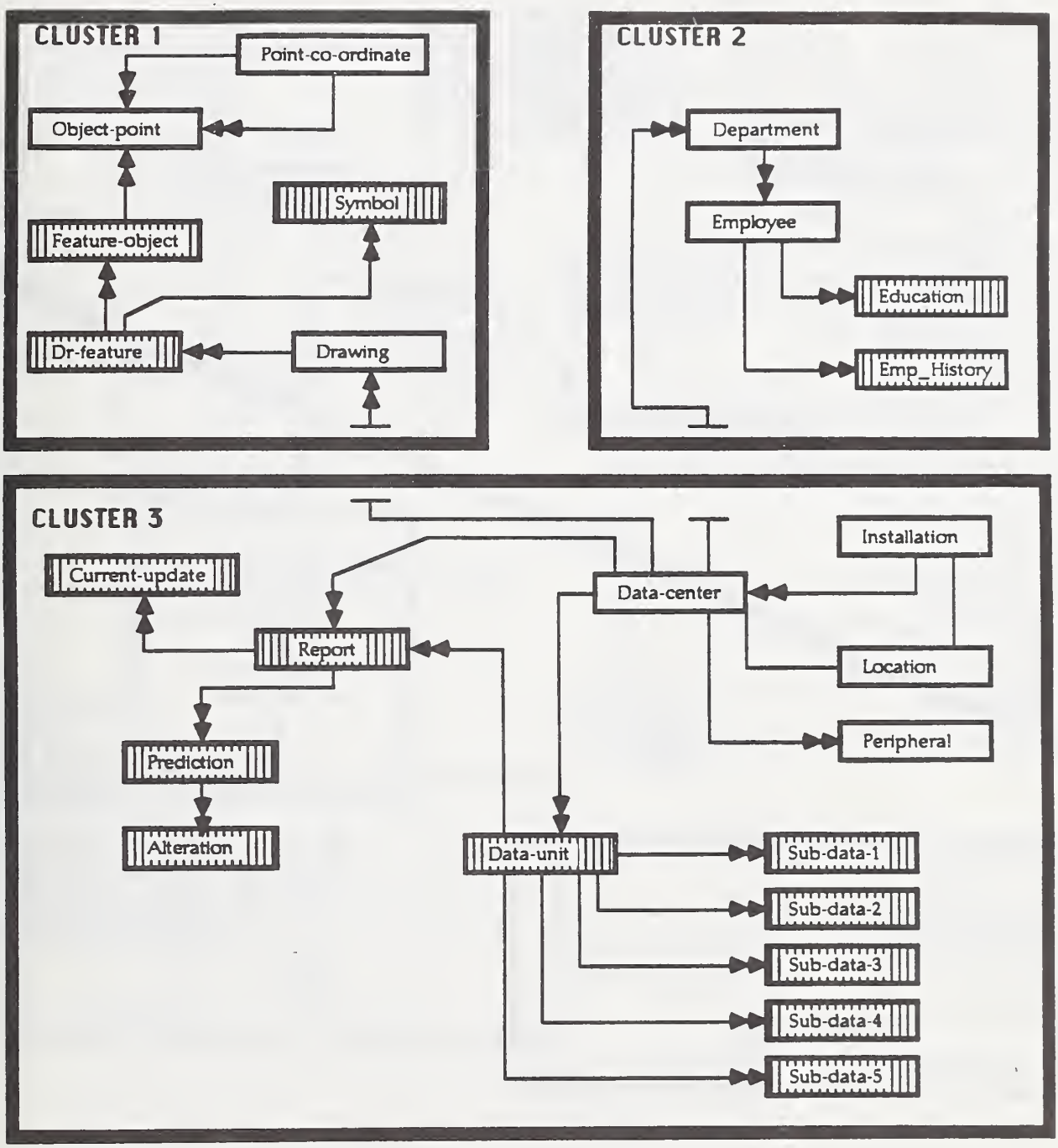


Figure 6. Division of Sample LDS into Three Clusters

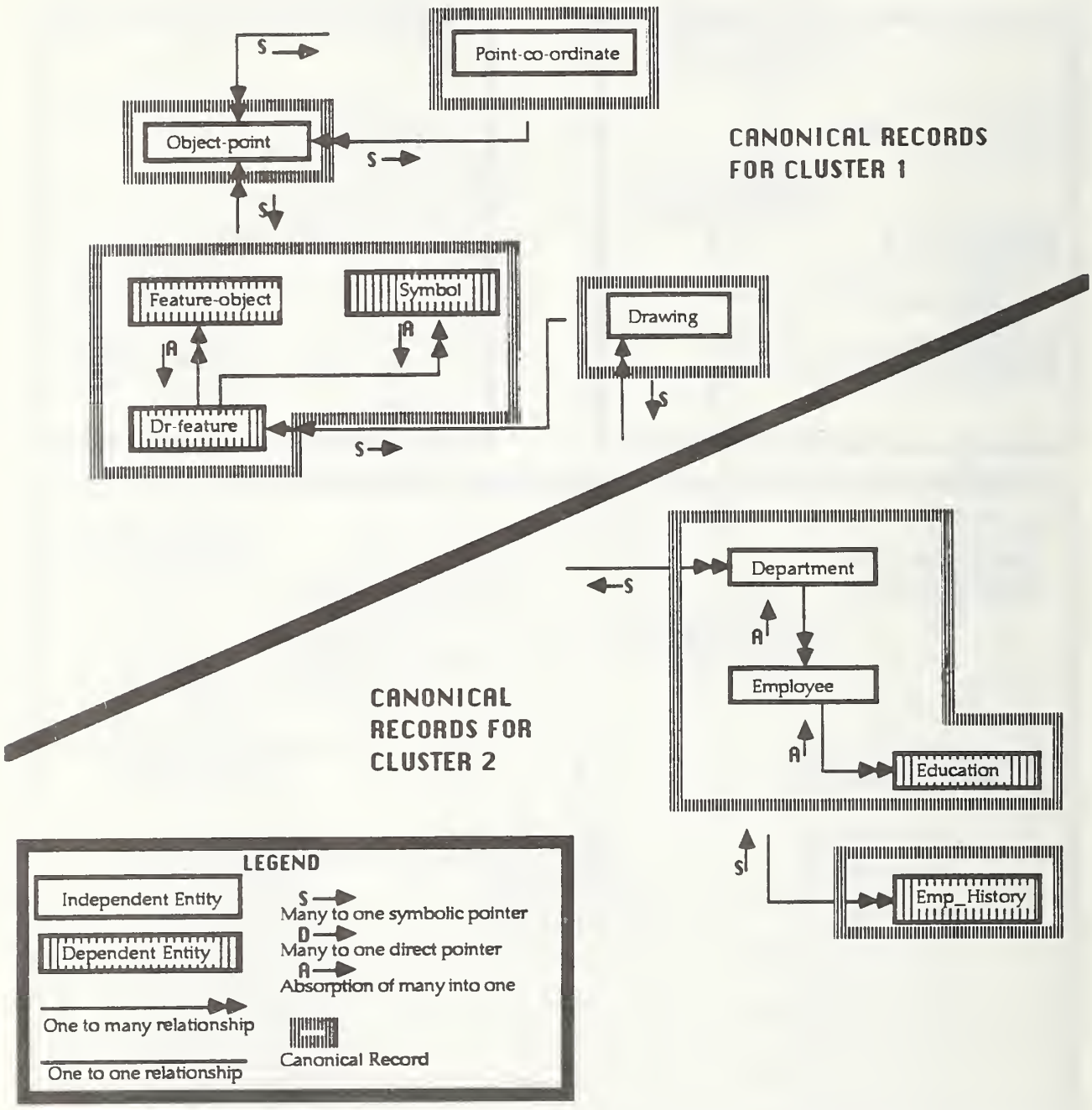


Figure 7. Canonical Records in Clusters 1 and 2

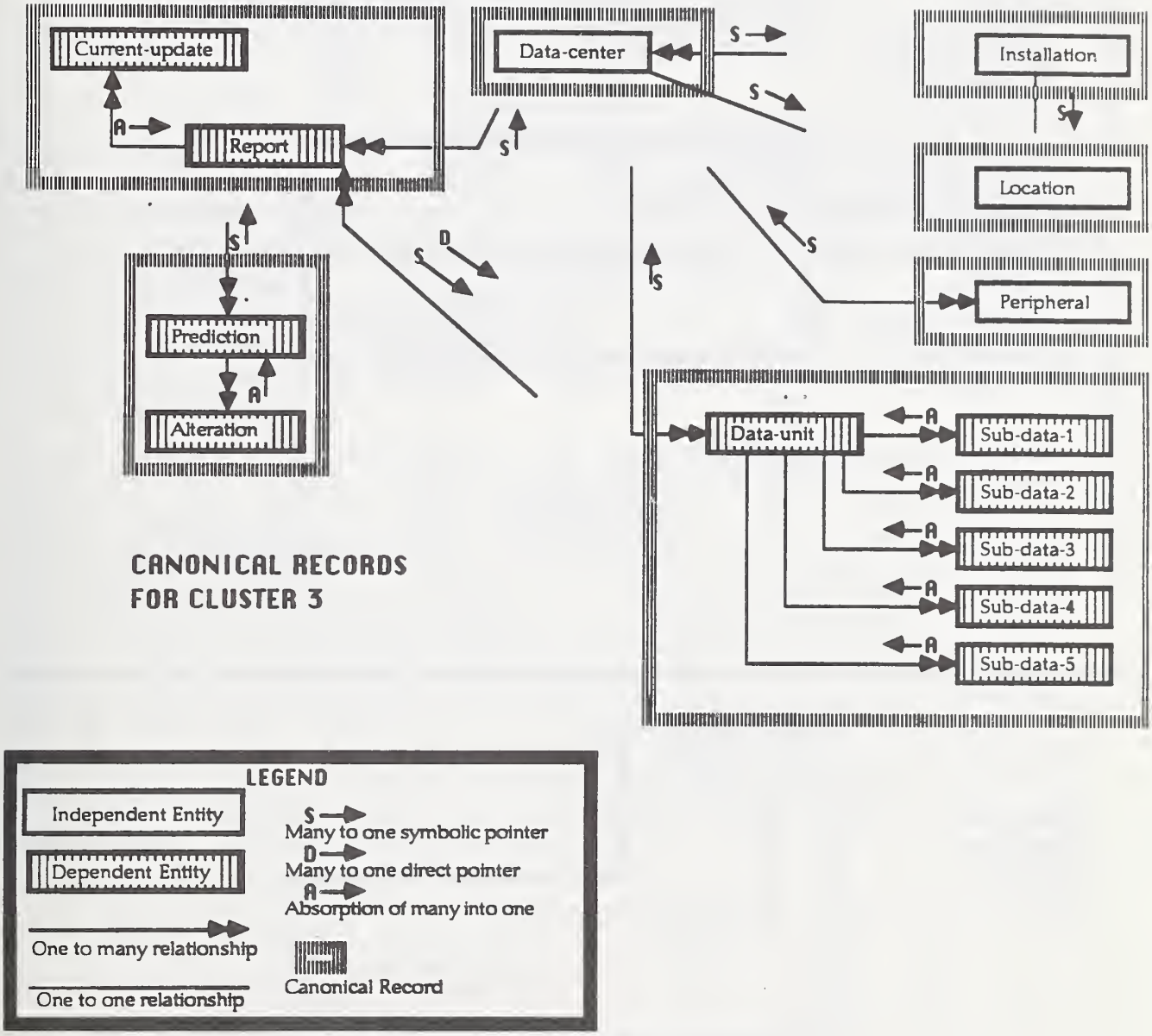
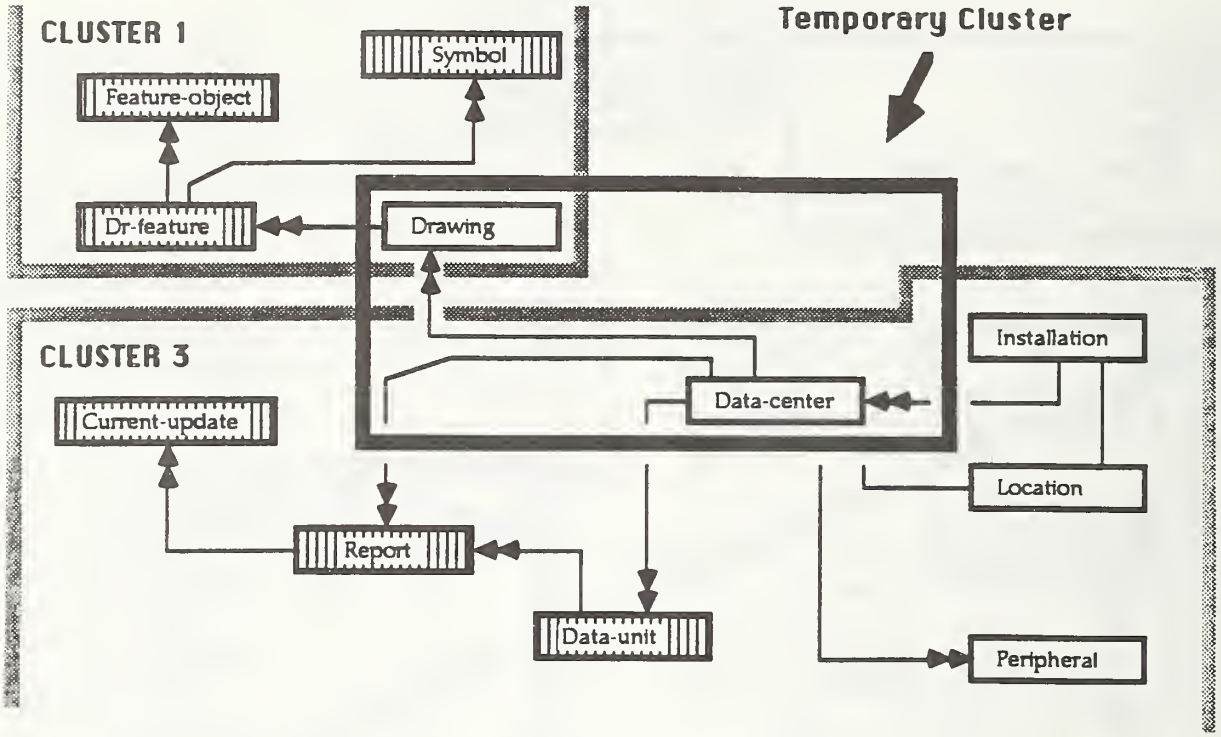


Figure 8. Canonical Records in Cluster 3



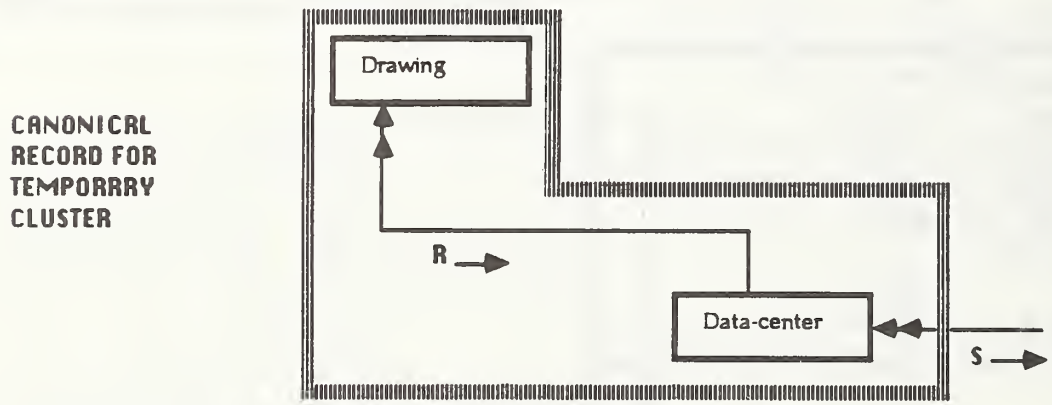


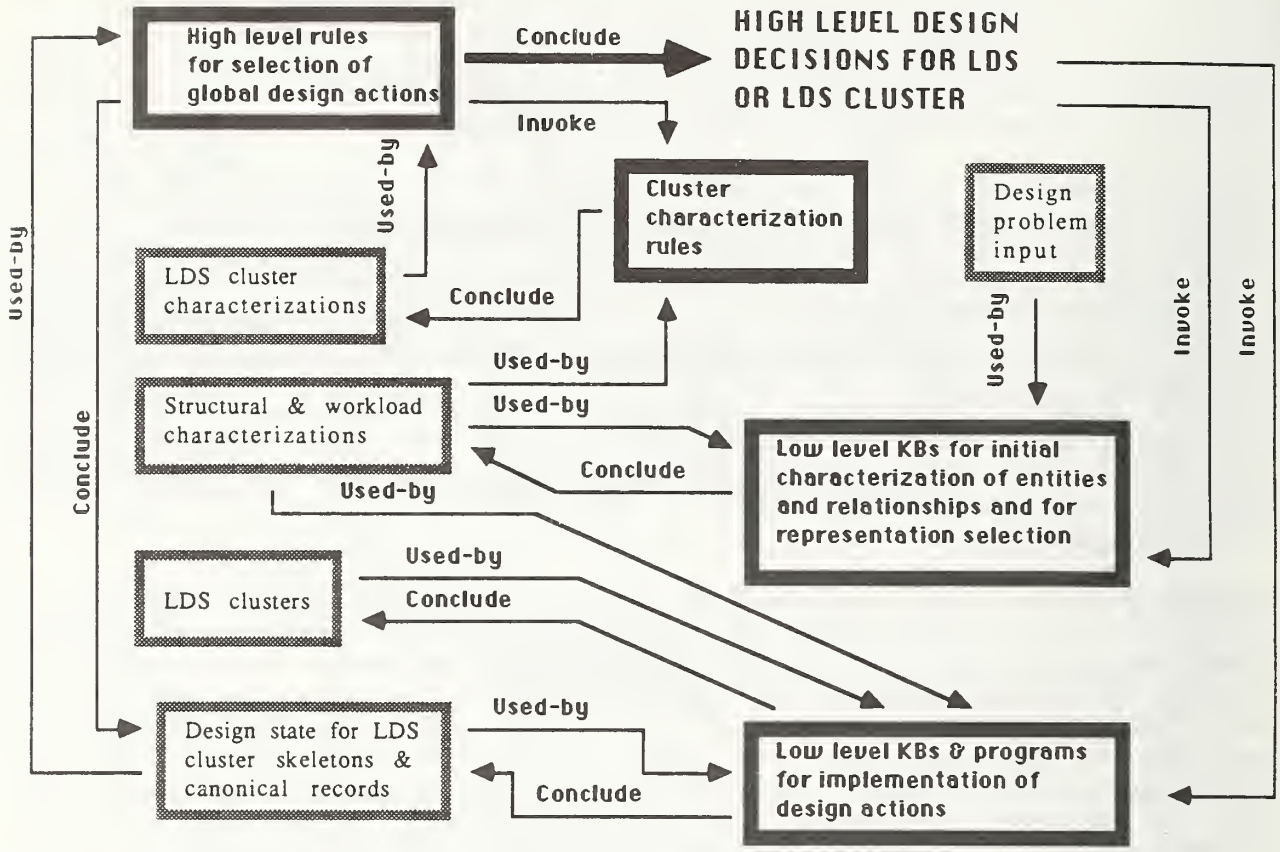
Figure 9. Temporary Cluster from Clusters 1 and 3

To select a cluster to work on and to determine the appropriate design action, the High Level must consider what design work has already been done on the cluster. High Level rules for making design decisions also rely on characterizations about LDS clusters. Cluster characterizations are statements about the size of the cluster and the amount of workload complexity in the cluster. Rules for concluding cluster characterizations rely on the initial characterization of entities and relationships.

Design actions may be predetermined or non-predetermined. Predetermined design actions must always be performed at a certain time or in a certain order. For instance, initial characterization of all entities and relationships in the LDS and initial selection of reasonable representations should take place first because subsequent actions depend on the information obtained from these characterizations. Most of the design actions taken by the High Level are non-predetermined. They are not performed in a specific order and are dependent on the individual characteristics of specific problems. Decisions relating to problem reduction and record formation are not predetermined.

Design actions are implemented by applying Low Level knowledge bases to the chosen LDS cluster and by invoking specialized programs which perform algorithmic computations. The KBS has several knowledge bases for this purpose. Figure 10 describes the flow of control initiated by a High Level design decision, and the relationship of the High Level to other parts of the knowledge-based system. The process by which knowledge bases are used to implement a design action is described in section 4.5.

Design actions continue until potentially low cost canonical records have been identified for each cluster and all relevant recombinations of clusters have taken place. At this point the High Level may terminate design activity and the selected canonical records can be submitted for fine-tuning by conventional algorithmic methods.



This diagram summarizes the control and information flow triggered by a High Level design decision. The **invoke** arrows refer to flow of control, **used-by** arrows indicate information needed for inferencing, and **conclude** arrows refer to conclusions made. An LDS cluster or clusters is chosen for a design action based on characterizations made by High Level characterization rules. Appropriate knowledge base(s) and/or routines (shown in thick boxes) are invoked to implement the chosen design action. Application of a knowledge base to a cluster results in referencing and possibly updating information about the design state of the cluster or clusters (shown in thin boxes). The High Level continues to specify design actions until the design state for the entire LDS is complete. Efficient canonical records are then selected and fine-tuned. Knowledge bases for initial characterizations and for initial selection of representations are invoked once at the beginning of the design process.

Figure 10. A Diagram of High Level Processes

4.2 DESIGN ACTIONS FOR PROBLEM REDUCTION

Design actions for problem reduction include 1) division of the LDS, or of a large LDS cluster, into smaller clusters, and 2) selective reduction of the number of relationship representations within clusters.

4.2.1 Dividing the LDS

The High Level contains rules which recommend division of the LDS or the division of any of the LDS clusters. The principle is to divide and conquer. Division is recommended if the LDS, or LDS cluster, is found to contain a large number of entities and relationships which can be processed in smaller separate clusters. The goal is to produce a few clusters of moderate size without excessive fragmentation, in which the amount of workload complexity within the divided clusters exceeds the workload complexity along the relationships which connect the clusters. See Figure 5 above.

Division of the LDS, or of a cluster, requires first selecting a specific division rule set. This rule set is then used to determine a set of breakpoints, that is, a set of relationships which subdivide the problem. For example, in Figure 5, if the relationship between DEPARTMENT and DATA-CENTER has very little activity, and the activity is in only one direction, it may serve as a breakpoint. This will result in the creation of the small, separate cluster in the upper right corner of Figure 5 (shown explicitly in Figure 6). The relationship itself will be fixed to one non-absorbing representation (note that absorption would require that DEPARTMENT and DATA-CENTER belong to the same cluster).

Division rule sets are based on predetermined criteria for breakpoint selection. The most restrictive (and therefore most reliable) rule set, which is least likely to separate heavily interacting entities, is tried first (section 4.5.4). The result of the division action is then reviewed. If the division is judged successful, the newly divided clusters become future subjects for design actions. If the original LDS was insufficiently broken up by the preceding action, that is, one or more clusters were produced which were too large, the High Level may repeat the action. In this case, the High Level will select a division rule set with less restrictive criteria and apply it to each large cluster. If the original LDS cannot be divided, e.g., the complexity of workload interrelationships between component entities and relationships is so strong that breakpoints cannot be found, or if use of the division rule set produces excessive fragmentation, the division action may be unsuccessful.

4.2.2 Direct Reduction of Problem Size

Restricting the number of relationship representations for the LDS, or for a cluster, is an alternative way to reduce problem size. This design action requires first identifying relationships which are characterized as having a high degree of structural and workload complexity, indicating that many alternative representations and resulting canonical records may need to be examined. The remaining relationships are then restricted to one representation having the highest certainty factor. This design action may be invoked by the High Level for clusters where attempts at division have failed, or for smaller clusters deemed less critical, and deserving of less design effort. The action results in the reduction of the number of resulting skeletons and canonical records, often to considerably more manageable levels.

4.3 DESIGN ACTIONS FOR FORMATION OF CANONICAL RECORDS

The High Level determines how to form canonical records based on the number of possible skeletons in the cluster. There are two different design actions for canonical record formation.

4.3.1 Enumeration of all Alternative Skeletons

For clusters having a small number of skeletons, the High Level may elect to enumerate the alternative skeletons and form all canonical records. The Cost Estimation Function (section 4.5.6) is a module within the KBS which accepts an individual skeleton together with itemized information about the workload for each entity and relationship in the cluster. The function returns the estimated total cost of processing the workload for the skeleton. As skeletons are enumerated, the Cost Estimation Function is applied to each skeleton. The skeletons are ranked by cost and a subset of low cost skeletons is selected. The canonical records of these skeletons are then fine-tuned. The High Level chooses this design action for small clusters which have first undergone relationship restriction, or for small clusters having few alternative skeletons. It is not appropriate for larger, more complex clusters.

4.3.2 Selective Generation of Records and Skeletons

For clusters with many skeletons, the effects of the combinatorial explosion are averted by selectively generating skeletons. The goal of this strategy is to identify a small number of low cost skeletons without completely enumerating all the alternatives. The canonical records of these skeletons are then considered good candidates for fine-tuning. The selective

generation of skeletons is a complex process which involves a generate and test strategy. The strategy is based on in-depth analysis of design alternatives for the LDS cluster, and uses cost estimates for alternative skeletons, canonical records, and relationship representations. The result of using this strategy is a search for low cost skeletons in a search space with many alternatives. The High Level controls the direction of the search by using an extensive Low Level knowledge base of rules. These rules contain complex analytical knowledge about structure, workload, and estimated performance of design structures. These rules are used to select alternative relationship representations resulting in the generation of new skeletons with new canonical records. The High Level applies this knowledge base in a systematic manner. The process is described in greater detail below in section 4.5.5 and 4.5.6. The appendix contains an example of skeleton generation for Cluster 2, shown in Figure 6.

4.4 RECOMBINATION OF DIVIDED CLUSTERS

Cluster recombination compensates, in part, for the shortcomings of dividing the LDS. When an LDS is divided into clusters, representations for relationships connecting different clusters are not varied, and a single non-absorbing representation for each connecting relationship is chosen. Representations for connecting relationships cannot be varied when doing work on an individual cluster because to do so would necessitate having to consider the two clusters together. This would result in having to work on a larger combined cluster, negating the benefits of having created the smaller clusters. However, maintaining separate clusters and not varying connecting relationships results in the possibility of overlooking better designs. This is especially so where less restrictive criteria for workload complexity were used in breakpoint selection. The use of less restrictive criteria results in clusters with connecting relationships which may have significant design problems. Such connecting relationships should have their representations varied.

In cluster recombination, a temporary cluster is created from portions of adjacent clusters. The temporary cluster includes the connecting relationships for the original clusters. Also, limited portions of the original clusters, consisting of estimated low cost canonical records adjoining the connecting relationships, are included in the temporary cluster. This new "hybrid" cluster then becomes the subject of further design activities. Refer to Figure 9 above. As stated previously, this figure shows the creation of a new temporary cluster from portions of Clusters 1 and 3 in Figure 6. The High Level is responsible for determining which adjacent clusters should be recombined, for initiating and controlling design actions to form canonical records in the temporary cluster, and for analyzing the

results of these actions by comparing the qualifying canonical records having a low estimated cost in the temporary cluster with the qualifying records of the original clusters. The result may lead to an adjustment in the qualifying records submitted for fine-tuning by a conventional algorithmic design system.

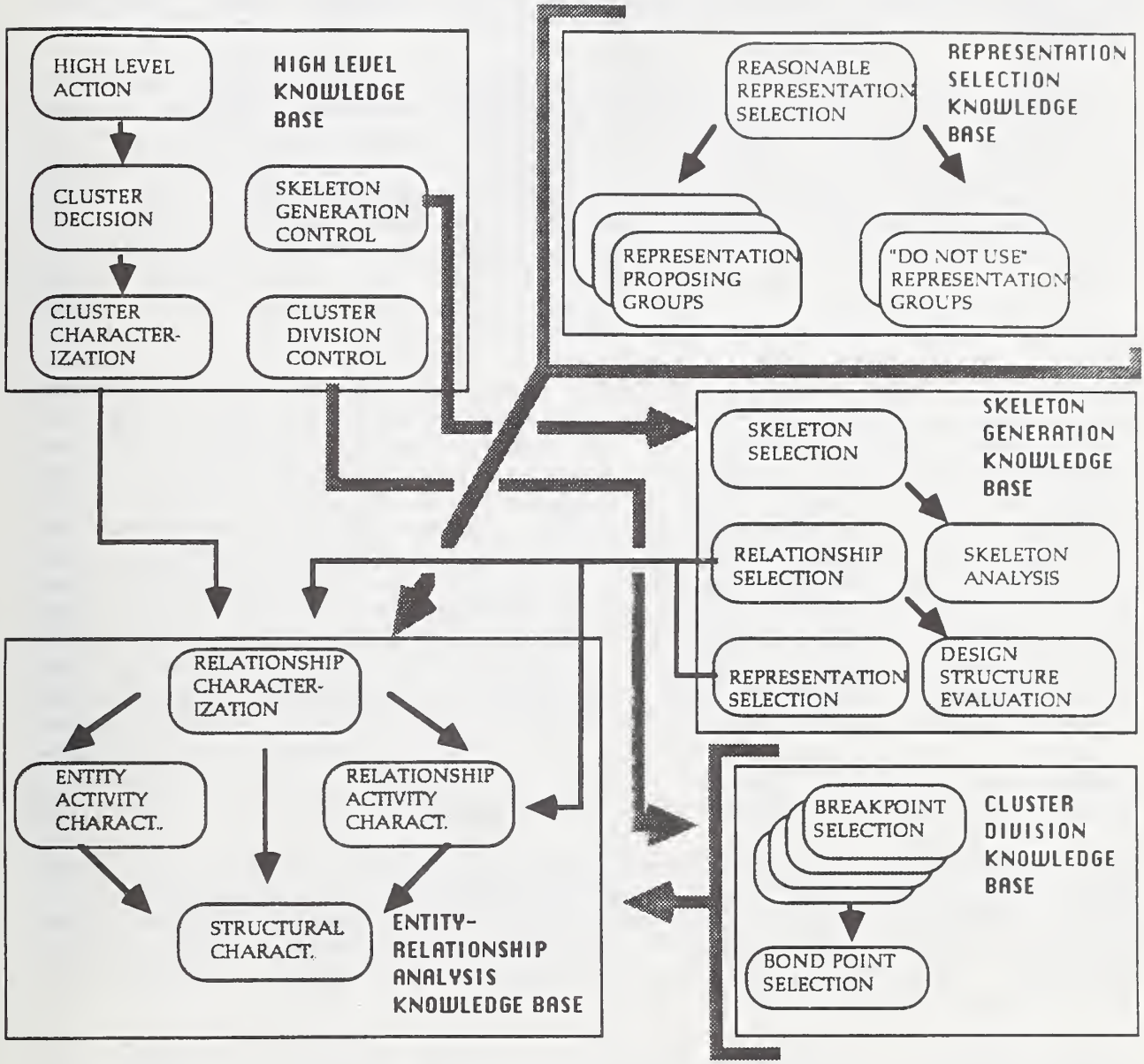
4.5 KNOWLEDGE BASES FOR IMPLEMENTATION OF DESIGN ACTIONS

Low Level knowledge bases are invoked to carry out design actions and to perform specialized tasks. These knowledge bases contain the problem solving expertise of human designers. Each knowledge base consists of one or more rule groups. Rule groups are collections of rules dedicated to making conclusions about specific subject areas or kinds of facts. They may be thought of as having specific tasks to accomplish, such as the conclusion of certain types of facts which are used by other parts of the knowledge-based system. For example, all rules which conclude relationship representations on the basis of workload characterizations belong to one rule group, while rules which conclude relationship representations based on structural characterizations belong to another rule group. The relationship representations concluded by these rule groups are facts used to carry out design actions for record formation (section 4.3).

Knowledge bases contain rule groups whose subject areas and tasks are similar. For example, both of the rule groups mentioned above belong to the same knowledge base. Knowledge bases and rule groups are intended to provide an organizational framework useful for knowledge engineers and domain experts in understanding and maintaining the knowledge-based system. Rule groups are modular units which can be independently understood and maintained. This allows knowledge engineers and domain experts to experiment with different rules, and to develop improved rule groups with enhanced problem solving capabilities.

Individual rule groups can be thought of as being dependent on each other for information. In other words, one rule group may conclude facts which rules in another rule group require in their IF parts. These interdependencies are extensive and cut across knowledge bases. Figure 11 shows the knowledge bases and rule groups and shows dependencies between individual rule groups.

Some rule groups rely on given facts about the LDS, the workload, or on facts computed by algorithmic routines. This is not shown in Figure 11. Some rule groups are dependent entirely on such information to make inferences and do not depend on other rule groups. For instance, the Skeleton Analysis and Design Structure Evaluation Rule Groups in the Skeleton Generation Knowledge Base rely only on the Cost Estimation Function for information (See section 4.5.5).



The arrows indicate the direction of the dependency. Thin arrows denote dependency of one rule group on another. The thick arrows denote dependency of a rule group or entire knowledge base on all the rule groups of another knowledge base.

Figure 11. Dependencies between Rule Groups

4.5.1 The High Level Knowledge Base

The High Level Knowledge Base is used to manage the processing of physical design. This knowledge base consists of five rule groups.

The High Level Action Rule Group is invoked to determine the next action to be taken on the physical design problem being processed. Actions chosen include the invocation of other High Level rule groups to determine decisions on clusters, invocation of individual rule groups in Low Level knowledge bases to carry out specific tasks such as initial characterization of entities and relationships, and direct management of Low Level activities such as cluster division and selective skeleton generation. The Action Rule Group is also responsible for determining when clusters should be recombined.

The Cluster Characterization Rule Group and the Cluster Decision Rule Group are invoked to determine design actions for a cluster. These High Level rule groups are invoked as a result of actions determined by the Action Rule Group. The Cluster Decision Rule Group is invoked each time a design decision must be made on a cluster. The Cluster Characterization Rule Group is invoked when a cluster is created. The Cluster Decision Rule Group depends on the Cluster Characterization Rule Group for information. The Cluster Characterization Rule Group in turn relies on information concluded by rule groups in the Entity-Relationship Analysis Knowledge Base during initial characterization of entities and relationships. The decisions made by the Cluster Decision Rule Group are used by the Action Rule Group to conclude actions for a cluster. This may include any of the actions for problem reduction, canonical record formation, as well as a determination that design work on a cluster is complete.

The Skeleton Generation Control Rule Group manages the activities of the Skeleton Generation Knowledge Base. This High Level rule group determines which rule group within the Skeleton Generation Knowledge Base to invoke to further the skeleton generation process. Skeleton generation is described in detail in sections 4.3.2 and 4.5.5.

The Cluster Division Control Rule Group manages the implementation of LDS division. The main purpose of this rule group is to select a rule set for breakpoint determination. This process will be described in sections 4.2.1 and 4.5.4.

4.5.2 The Entity-Relationship Analysis Knowledge Base

This knowledge base is used for individual characterizations of entities, relationships, and small combinations of entities and relationships. This knowledge base contains rules which capture analytical knowledge for making simple characterizations of the structure and workload for individual entities and relationships. These simple characterizations are combined into complex characterizations about individual entities, relationships, and small aggregations of entities and relationships. Figure 12 shows an example using a simplified version of the actual rules. The actual knowledge base is quite large and contains a variety of analytical information about relationships and patterns based on workload.

The Entity-Relationship Knowledge Base consists of four rule groups. The Structural Characterization Rule Group makes characterizations of relationship degree, entity type, and entity dependency for individual entities and relationships. The Entity Activity Rule Group concludes similar characterizations about direct activity on individual entities. The Relationship Activity Rule Group is responsible for conclusions about activity levels along relationships. The Relationship Characterization Rule Group makes conclusions about workload complexity for individual relationships and for small groups of relationships. The Relationship Characterization Rule Group depends on the Relationship Activity Rule Group, the Entity Activity Rule Group, and the Structural Characterization Rule Group. Figure 12 contains examples of rules from the Structural Characterization, Relationship Activity Characterization, and Relationship Characterization Rule Groups.

The rules of this knowledge base are invoked once by the High Level at the beginning of a session, and are applied to the entire LDS. This action constitutes the initial characterization of entities and relationships referred to above. The results are stored internally and provide a basis for decisions made by other parts of the system in subsequent design work.

```
{RULE A}   (Structural Characterization Rule Group)
IF  RELATIONSHIP  ?Rel-id ?Ent1 ?Ent2 ?Rel-name
    PRIMARY-IDENTIFIER  ?Ent2 ?Identifier
    EQUAL           ?Rel-name ?Identifier
    THERE-IS --> ATTRIBUTE  ?Ent2 ?Att-name NON-IDENTIFIER
THEN
    DEPENDENT-ON    ?Ent2 ?Ent1


{RULE B}   (Relationship Activity Char. Rule Group)
IF  REL-ACTIVITY  ?Rel-id ?Ent1 ?Ent2
                  ?Subset-size ?Rel-frequency
    ENTITY-ACTIVITY ?Ent1 ?Ent1-frequency
    ENTITY-ACTIVITY ?Ent2 ?Ent2-frequency
    GREATER-THAN    ?Rel-frequency 1/4(SUM ?Ent1-frequency
                                           ?Ent2-frequency)
THEN
    CHARACTERIZATION ?Rel-id ?Ent1 ?Ent2 HIGH-ACTIVITY


{RULE C}   (Relationship Characterization Rule Group)
IF  CHARACTERIZATION ?Rel-id ?Ent1 ?Ent2 HIGH-ACTIVITY
    CHARACTERIZATION ?Rel-id ?Ent2 ?Ent1 HIGH-ACTIVITY
THEN
    POSSIBLE-DESIGN-PROBLEM ?Rel-id HIGH-BI-DIRECTIONAL-ACTIVITY
```

If we substitute REL_2 for ?Rel-id, EMPLOYEE for ?Ent1, EDUCATION for ?Ent2, EDUCATION-OF-EMPLOYEE for ?Rel-name, and EDUCATION-OF-EMPLOYEE for ?Identifier, {RULE A} characterizes EDUCATION as a dependent entity. {RULE B} states that a relationship should be characterized as having a high activity level from one entity to another if the frequency of this activity exceeds one quarter of the frequency of both entities combined. Making the same substitutions, this characterization can be applied to the activity between EMPLOYEE and EDUCATION along REL_2.

{RULE B} could be applied twice to characterize a relationship as having high activity in both directions. {RULE C} could then combine both characterizations into a complex characterization identifying a relationship with high activity in both directions. {RULE C} is a simple example of a complex characterization based on simpler characterizations. Relationships to which it applies might then be designated problem areas which require greater attention.

Figure 12. Rules for making Characterizations

4.5.3 The Representation Selection Knowledge Base

The purpose of the Representation Selection Knowledge Base is to determine a set of reasonable representations for all relationships in the LDS. This knowledge base relies heavily on the characterizations made by the rules in the Entity-Relationship Knowledge Base. The rules in this knowledge base use characterizations about structure and workload to 1) propose possible representations with an associated strength of belief represented by a certainty factor, and 2) determine that a particular representation should not be used, also with an associated strength of belief. As is the case in the example of the human designer described previously, several different possible representations may be concluded for the same relationship. Similarly, one or more representations may be determined to be poor choices for a relationship. Both types of determinations may be made for the same relationship representation. More than one rule may execute to make a positive or negative conclusion about a representation. Each rule corresponds to a different reason for the conclusion. Six rule groups from this knowledge base will be described.

The knowledge base has three rule groups for proposing possible representations. One rule group proposes simple symbolic pointers, direct pointers, and absorption based on structural characteristics. A second rule group proposes the same representations based on entity and relationship activity characterizations. A third rule group proposes more complex representations not covered by the previous rule groups. These representations involve combinations of symbolic and direct pointers, and are used only in unusual circumstances. All three rule groups are dependent on rule groups in the Entity-Relationship Analysis Knowledge Base for information.

There are two rule groups to determine which representations should not be used. One rule group makes negative determinations based on structural characterizations. The other rule group uses workload characterization information. These rule groups also depend on the Entity-Relationship Analysis Knowledge Base.

A sixth rule group combines the conclusions of the previous five rule groups, using their certainty factors to conclude reasonable representations. The certainty factor associated with this combined determination is based on the positive certainty factor of the possible representation together with a negative certainty factor associated with any determinations that the representation should not be used. Those relationship representations which have a certainty factor of sufficient strength are regarded as reasonable representations which may be used as alternatives in the formation of skeletons and canonical records. Refer to Figure 13 for a brief example.

```
{RULE D}  (Structural Proposing Rule Group)
IF  RELATIONSHIP ?Rel-id ?Ent1 ?Ent2 ?Rel-name
    DEPENDENT-ON ?Ent2 ?Ent1
THEN
    POSSIBLE REPRESENTATION ?Ent1 ABSORBS ?Ent2
    CERTAINTY FACTOR 0.5

{RULE E}  (Structural Proposing Rule Group)
IF  RELATIONSHIP ?Rel-id ?Ent1 ?Ent2 ?Rel-name
    DEGREE ?Rel-id ?Ent1 to ?Ent2 is 1 to M
THEN
    POSSIBLE REPRESENTATION ?Ent2 SYMBOLIC-POINTS-TO ?Ent1
    CERTAINTY FACTOR 0.5

{RULE F}  (Activity Proposing Rule Group)
IF  DEGREE ?Rel-id ?Ent1 to ?Ent2 is 1 to M
    CHARACTERIZATION ?Rel ?Ent1 ?Ent2 HIGH-ACTIVITY
THEN
    POSSIBLE REPRESENTATION ?Ent1 ABSORBS ?Ent2
    CERTAINTY FACTOR 0.25

{RULE G}  (Activity Anti-Proposing Rule Group)
IF  CHARACTERIZATION ?Rel-id ?Ent1 ?Ent2 HIGH-ACTIVITY
    DEPENDENT-ON ?Ent2 ?Ent1
    ABSENT --> DEPENDENT-ON ?Any-entity ?Ent2
THEN
    PREVENTS REPRESENTATION ?Ent2 SYMBOLIC-POINTS-TO ?Ent1
    CERTAINTY FACTOR 0.25
```

This example continues from that in Figure 12. EDUCATION is dependent on EMPLOYEE, so {RULE D} proposes absorption as a possible representation. {RULE E} proposes an M to 1 symbolic pointer for the same reason. Absorption is also recommended by {RULE F} based on a high activity level along REL_2.

Using the Bernoulli method, we combine the certainty factors for {RULE D} (0.5), and {RULE F} (0.25) to obtain a combined certainty for absorption of 0.625 (0.625 = 0.5 + 0.25 * (1.0-0.5)). {RULE E} provides a single recommendation for an M to 1 symbolic pointer with a certainty of 0.5. The certainty of using a symbolic pointer is decreased to 0.25 (0.25 = 0.5 - 0.25) by {RULE G} which provides a negative certainty factor. This rule recommends against using an M to 1 symbolic pointer if there is high retrieval activity in the 1 to M direction, if the "M" entity depends on the "1" entity, and if no third entity depends on the "M" entity. Although absorption has a higher associated certainty, both representations might be attempted by the KBS.

Figure 13. Rules for Selection of Representations

The knowledge base incorporates the expertise of the human designer in selecting relationship representations and in representing the effect of different and often conflicting structural and workload considerations. Just as in the case of the Entity-Relationship Analysis Knowledge Base, this knowledge base is applied by the High Level at the beginning of each session and is responsible for determining an initial set of reasonable representations for the entire LDS.

4.5.4 The Cluster Division Knowledge Base

The Cluster Division Knowledge Base is used to select relationships to serve as breakpoints in an LDS. This knowledge base contains rules which use structural and workload characterizations to recommend relationships to serve as breakpoints between clusters of the LDS. The objective is to create LDS clusters having greater internal workload complexity than that of the relationships which connect them. The rules are divided into four Breakpoint Selection Rule Groups based on the degree of restrictiveness in the criteria used to make selections. Note that using less restrictive criteria will generate more breakpoints while using more restrictive criteria will generate fewer breakpoints. If more breakpoints are generated, the problem becomes easier to deal with. This is because generating more breakpoints generally produces many small clusters, each having fewer alternative skeletons to examine. However, because the relationships chosen as breakpoints will be fixed to one representation, the solution may be less satisfactory since promising alternatives involving the breakpoint relationships will not be explored. The solution to this problem is provided by recombining clusters, described in section 4.4.

This knowledge base has four rule groups ordered by the degree of restrictiveness in their breakpoint selection criteria. The most restrictive rule group is first followed by rule groups based on successively more relaxed constraints. The restriction criteria themselves are based on the conclusions of a fifth rule group, known as the Bond Point Rule Group. The Bond Point Rule Group exists for determining reasons that relationships should not serve as breakpoints. Relationships may be selected as bond points on the basis of characterizations of workload complexity. The level of restrictiveness associated with a Breakpoint Selection Rule Group is determined by combinations of bond point conclusions. All of the rules in a Breakpoint Selection Rule Group then contain the same restrictiveness criteria in their IF parts. See Figure 14. The rules of the Bond Point Rule Group are based on the workload characterizations made by the Entity-Relationship Analysis Knowledge Base invoked at the beginning of the session.

```
{RULE H}
IF  POSSIBLE-DESIGN-PROBLEM ?Rel-id HIGH-BI-DIRECTIONAL-ACTIVITY
THEN
    BOND-POINT ?Rel-id HIGH-BI-DIRECTIONAL-ACTIVITY


{RULE I}
IF  POSSIBLE-DESIGN-PROBLEM ?Rel-id HIGH-LARGE-SUBSET-RETRIEVAL
THEN
    BOND-POINT ?Rel-id HIGH-LARGE-SUBSET-RETRIEVAL


{RULE J}
IF  POSSIBLE-DESIGN-PROBLEM ?Rel-id HIGH-NON-FORWARDING-ACTIVITY
THEN
    BOND-POINT ?Rel-id HIGH-NON-FORWARDING-ACTIVITY


{RULE K}
IF  SELECTED-BREAK-RULE-GROUP RESTRICTION-LEVEL-1
    RELATIONSHIP ?Rel-id ?Ent1 ?Ent2 ?Rel-name
    NOT << BOND-POINT ?Rel-id HIGH-NON-FORWARDING-ACTIVITY >>
    NOT << BOND-POINT ?Rel-id HIGH-LARGE-SUBSET-RETRIEVAL >>
    NOT << BOND-POINT ?Rel-id HIGH-BI-DIRECTIONAL-ACTIVITY >>
    CHARACTERIZATION ?Rel-id LOW-PERCENTAGE-FORWARDED-ACTIVITY
THEN
    BREAK-POINT ?Rel-id RESTRICTION-LEVEL-1
```

The rules in this figure illustrate the process by which breakpoints are selected. {RULE H}, {RULE I}, and {RULE J} are examples of rules in the Bond Point Rule Group. They identify relationships as bond points if they contain characterizations of workload complexity. HIGH-NON-FORWARDING-ACTIVITY refers to relationship activity which need not require traversal of the relationship. {RULE K} belongs to a Breakpoint Selection Rule Group having a certain restriction level. The restrictiveness criterion associated with this level is given by the negation of the three types of bond points in the IF part of the rule. The sixth clause in the IF part specifies the unique condition associated with this particular breakpoint rule: the relationship activity characterization of LOW-PERCENTAGE-FORWARDED-ACTIVITY. This means that a low percentage of the activity focusing on ?Ent1 is forwarded along the relationship ?Rel-id.

Figure 14. Rules for Selection of Breakpoints

When attempting to divide the LDS or a large LDS cluster, the breakpoint selection rule groups are applied in the order of most restrictive criteria. The objective is to divide the LDS using breakpoints having as little workload complexity as possible. The selection of which breakpoint selection rule group to use is controlled by the High Level.

4.5.5 The Skeleton Generation Knowledge Base

The Skeleton Generation Knowledge Base is used to generate the estimated best skeletons and canonical records in a cluster. This knowledge base consists of five major rule groups. Skeleton generation begins with the creation of an initial skeleton in which relationship representations are determined on the basis of certainty factors. The High Level then systematically invokes the different rule groups within this knowledge base to selectively vary relationship representations producing new skeletons with new canonical records.

The knowledge base relies, in part, on information provided by the Cost Estimation Function. This function computes the total cost of a skeleton based on the cost of processing the workload for each entity, relationship, and canonical record in the skeleton. In addition, the function provides an itemized breakdown of the cost of each entity, relationship and record (section 4.5.6).

The Skeleton Selection Rule Group uses cost information to identify low cost skeletons to be processed for further design activity. Skeletons which have a high cost, and those which lack promising design alternatives, are not selected. This allows concentration of design activity on the best skeletons containing the best canonical records and permits reduction of the search space by pruning more costly skeletons with inefficient records.

In general, a single "least cost" skeleton containing relationships which can be varied will be selected for further work from a group of one or more low cost skeletons. Once this skeleton has been identified, it must be decided which relationships within the selected skeleton should have their representations altered.

The Relationship Selection Rule Group determines which relationship representations must be altered. The determination is made on basis of structural characteristics, workload characteristics, and itemized information about the costs of accessing individual entities and relationships. The rules of the Relationship Selection Rule Group combine this information to identify one or more relationship representations which may be changed. This allows concentration of design activity on specific parts of the skeleton which require more effort. The

selection of relationships to be changed is based on high estimated workload cost or based on the existence of special workload problems. The Relationship Selection Rule Group also depends on a third rule group, the Design Structure Evaluation Rule Group, for identification of efficient canonical records and relationship representations from previously generated skeletons. Since efficient physical structures can be kept in place and not modified, the number of relationship representations which must be selected and varied is reduced as more skeletons are generated and low cost structures discovered.

After relationship selection, the Representation Selection Rule Group chooses alternative representations for the selected relationships. Then, a new skeleton containing a modified set of canonical records is generated together with a set of itemized costs provided by the Cost Estimation Function. The new skeleton and its records are added to the internal list of skeletons for the LDS cluster.

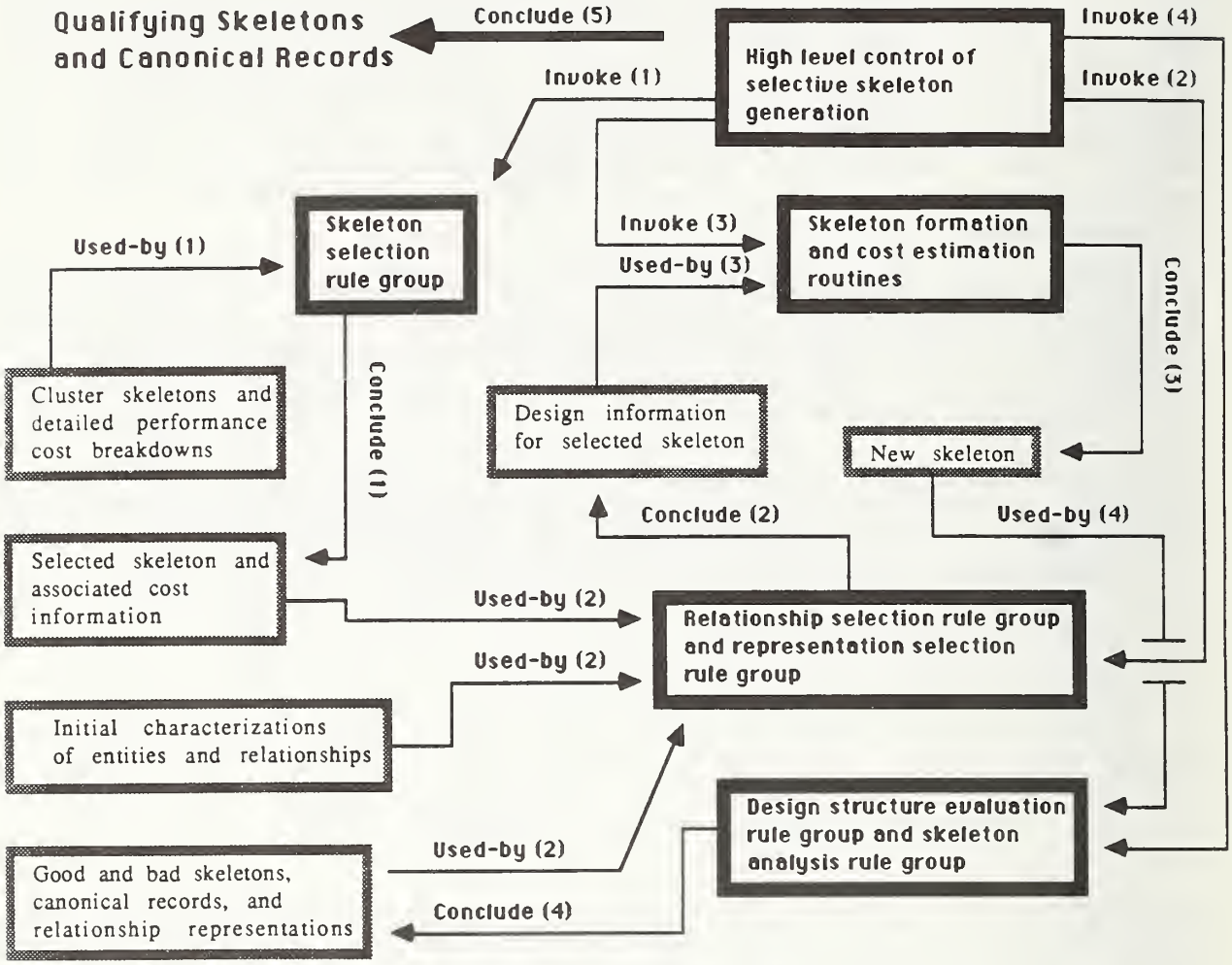
The Skeleton Analysis Rule Group exists to analyze each new skeleton on the basis of its estimated cost. If the new skeleton costs significantly more, it may be marked for pruning from the search space. If it is more efficient, then it is marked as a skeleton for future design efforts. The itemized cost breakdown is used by the Design Structure Evaluation Rule Group for identifying efficient and inefficient canonical records, and for identifying efficient and inefficient individual relationship representations. The Design Structure Evaluation Rule Group is invoked after the new skeleton is generated. The information concluded by this rule group will be used when varying relationship representations in subsequent actions on new skeletons.

At this point, the Skeleton Selection Rule Group may be invoked again to choose a new skeleton to work on. Or, if a sufficient number of skeletons have been generated, the High Level may terminate work on the cluster. Once work on the cluster ends, the most efficient skeletons are retained. Their canonical records may then submitted to a conventional algorithmic system for fine-tuning.

Figure 15 is a sketch of the selective skeleton generation process. The High Level selects a skeleton to be worked on from the list of skeletons for the cluster (1) (Skeleton Selection Rule Group). Selection of relationships and alternative representations to vary is made using cost information, initial characterizations, and previously identified efficient and inefficient canonical records. These conclusions are stored with the design information for the current skeleton (2) (Relationship Selection Rule Group and Representation Selection Rule Group). A new skeleton is generated using the altered relationship representations and the cost estimation routine is invoked (3).

The High Level now directs that the new skeleton be evaluated. Efficient and inefficient relationship representations and canonical records, if any, are also identified (4) (Design Structure Evaluation Rule Group and Skeleton Analysis Rule Group). The new skeleton is added to the cluster skeleton list. The High Level may then repeat the first step and again invoke the Skeleton Selection Rule Group to select a new skeleton for further work (possibly the new skeleton just created), or may terminate work on the cluster and select low cost skeletons and submit their canonical records for fine-tuning (5). See the appendix for an example of selective skeleton generation applied to Cluster 2.

The effect of the application of this knowledge base by the High Level is a search for low cost skeletons within a large search space. It is accomplished by generating a number of alternative designs in a generate and test fashion. A potential problem with this approach is the possibility of finding false minima, e.g., the generation of a sequence of skeletons forming a search path to a local minimum which may be greatly inferior to a minimum on another, unexplored search path. This problem can to some extent be handled by marking skeletons at branch points in the search tree and later returning to further explore alternatives.



This diagram summarizes the control and information flow during the selective generation of skeletons. The **invoke** arrows refer to transfer of control, **used-by** arrows indicate information needed for inferencing, and **conclude** arrows refer to conclusions that have been made. Knowledge bases and routines are shown in thick boxes. Facts and conclusions appear in thin boxes.

Figure 15. Selective Skeleton Generation

4.5.6 The Cost Estimation Function

A brief overview of the this function is required since it is integral to skeleton enumeration and selective skeleton generation. The function is responsible for modeling the cost of a particular skeleton and the costs of the canonical records within the skeleton. As stated previously, the costs are based entirely on the estimated cost of processing the retrieval and update workload. As each skeleton is generated (whether by enumeration or skeleton generation), the Cost Estimation Function is applied. Not only does this function produce a total cost for the skeleton and for each canonical record, but a cost estimate breakdown is also provided for each retrieval and/or update action for each relationship representation in the skeleton. Such a detailed breakdown is useful during skeleton generation when attempting to identify high cost relationship representations to vary.

The Cost Estimation Function relies on the 90/10 rule to produce a rough segmentation for a record; i.e., the most heavily used attributes, representing 90% of the workload, are assigned to the primary segment. A simplified access method selection algorithm is also used to model possible index structures. The Cost Estimation Function is not meant to be a substitute for more rigorous file organization design programs such as [MARC78] but instead is meant to supply information about probable costs of file organizations resulting from design structures chosen by the knowledge-based system.

5. CONCLUDING REMARKS

This report has described an approach taken in the design and implementation of a knowledge-based system for physical database design of large, complex, multi-entity logical data structures. To date, the system has been tested on several medium to large scale problems and, according to domain experts, has produced reasonable results. We anticipate testing the system using database management systems containing actual data in 1988. It is expected that this will result in the further development of the knowledge bases in the system.

Meanwhile, further work is continuing and is aimed at enhancing the individual knowledge bases and increasing their capacity to handle more types of design problems. Our long term goal is to have an in-house knowledge-based system capable of doing physical database design on a wide variety of problems simulating real-world hardware and software systems. In doing this, we hope to learn more about how to do physical database design and about the capabilities and uses of knowledge-based systems.

APPENDIX - AN EXAMPLE OF SELECTIVE SKELETON GENERATION

This appendix will describe the processing of Cluster 2, shown in Figures 1 and 6 and reproduced below in Figure 16, with workload provided in Figure 2. A narrative summarization of the sequence of steps undertaken by the system will be provided. A complete description of all the rules which were executed would be beyond the scope of this report.

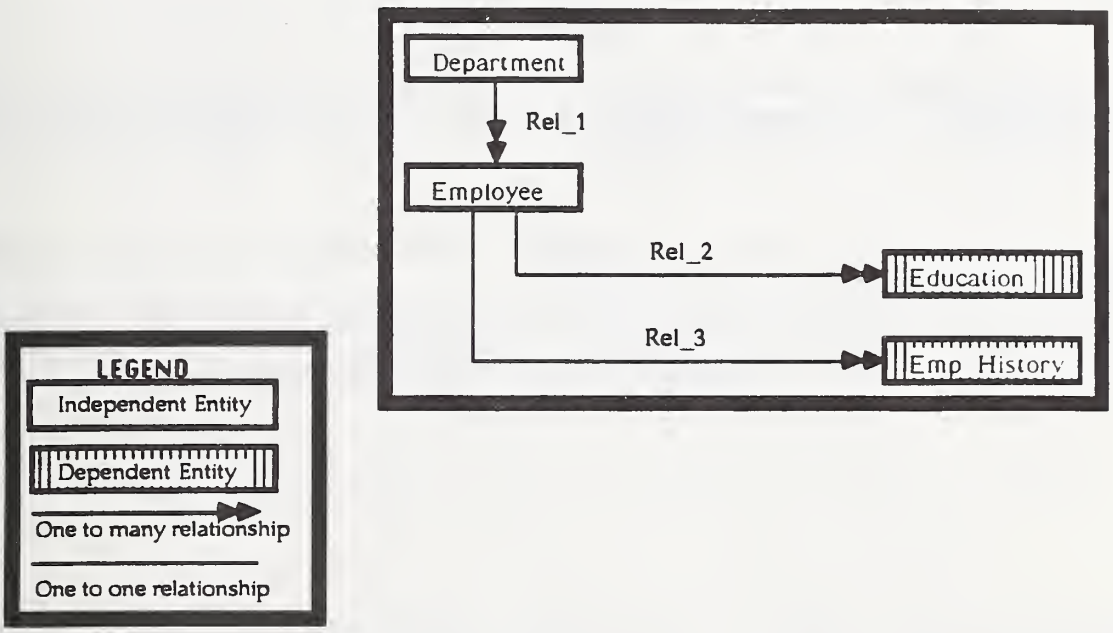


Figure 16. Cluster 2 from Figure 6

The example illustrates the cumulative effect of the Skeleton Generation Knowledge Base. For the purposes of the example, let us assume the information for entity lengths, entity cardinality, and relationship degree shown in Figure 17. Let us also assume that the initial set of reasonable representations is the set shown in Figure 18. REL_1 has three selected alternative representations, and REL_2 and REL_3 each have two, for a total of (3 * 2 * 2) = 12 skeletons. Actual clusters are typically much larger and contain many more relationships, reasonable representations, and alternative skeletons.

	Cardinality	Total Length	Primary Segment
DEPARTMENT	100	110	60
EMPLOYEE	3000	68	68
EDUCATION	6000	12	12
EMP_HISTORY	15000	252	52

	Degree
REL_1 DEPARTMENT to EMPLOYEE	-- 1 to 30
REL_2 EMPLOYEE to EDUCATION	-- 1 to 2
REL_3 EMPLOYEE to EMP_HISTORY	-- 1 to 5

Note: segmentation, and therefore the length of the primary segment, is based on a 90/10 rule.

Figure 17. Data on Entities and Relationships in Example

REL_1

```
REASONABLE REPRESENTATION REL_1 DEPARTMENT absorbs EMPLOYEE
REASONABLE REPRESENTATION REL_1 EMPLOYEE  symbolic pointer to
                                DEPARTMENT
REASONABLE REPRESENTATION REL_1 EMPLOYEE  direct pointer to
                                DEPARTMENT
```

REL_2

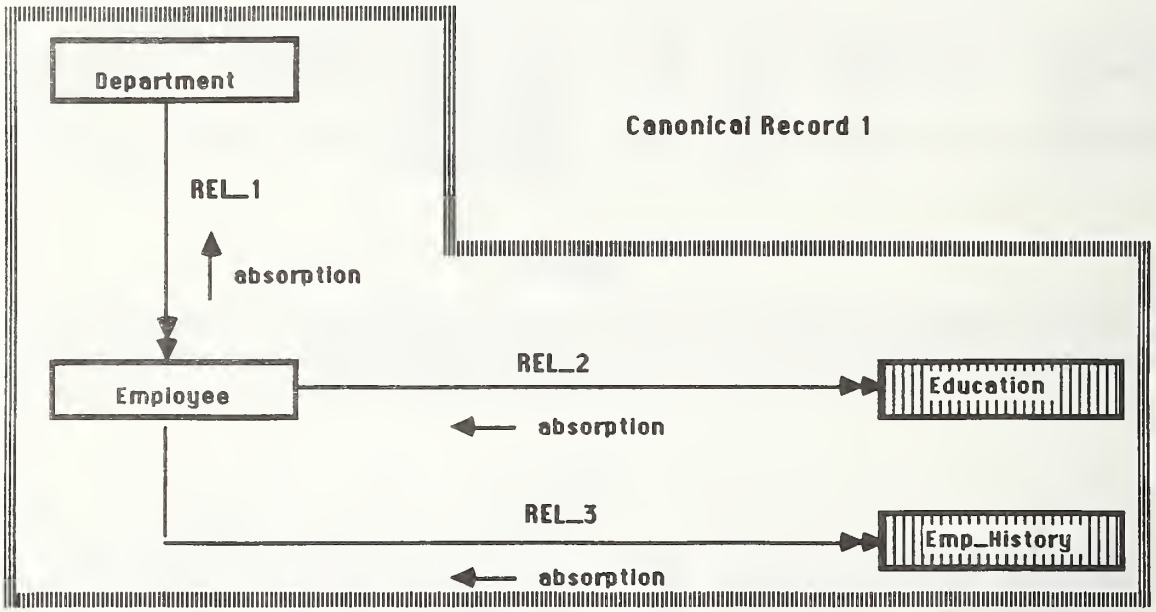
```
REASONABLE REPRESENTATION REL_2 EMPLOYEE absorbs EDUCATION
REASONABLE REPRESENTATION REL_2 EDUCATION symbolic pointer to
                                EMPLOYEE
```

REL_3

```
REASONABLE REPRESENTATION REL_3 EMPLOYEE absorbs EMP_HISTORY
REASONABLE REPRESENTATION REL_3 EDUCATION symbolic pointer to
                                EMP_HISTORY
```

Figure 18. An Initial Set of Reasonable Representations

Figures 19 and 20 represent a snapshot of design work in progress on the cluster. Two skeletons have already been generated, Skeleton 1 followed by Skeleton 2. Itemized cost estimates have been computed by the cost estimation algorithm and are shown. In Skeleton 1, EDUCATION and EMP_HISTORY are absorbed into EMPLOYEE, which in turn is absorbed into DEPARTMENT, creating Canonical Record 1, a single three-level hierarchical canonical record. In Skeleton 2, there are two canonical records. DEPARTMENT is the single entity in Canonical Record 2 while EDUCATION and EMP_HISTORY are absorbed into EMPLOYEE forming Canonical Record 3, a two-level hierarchical record.

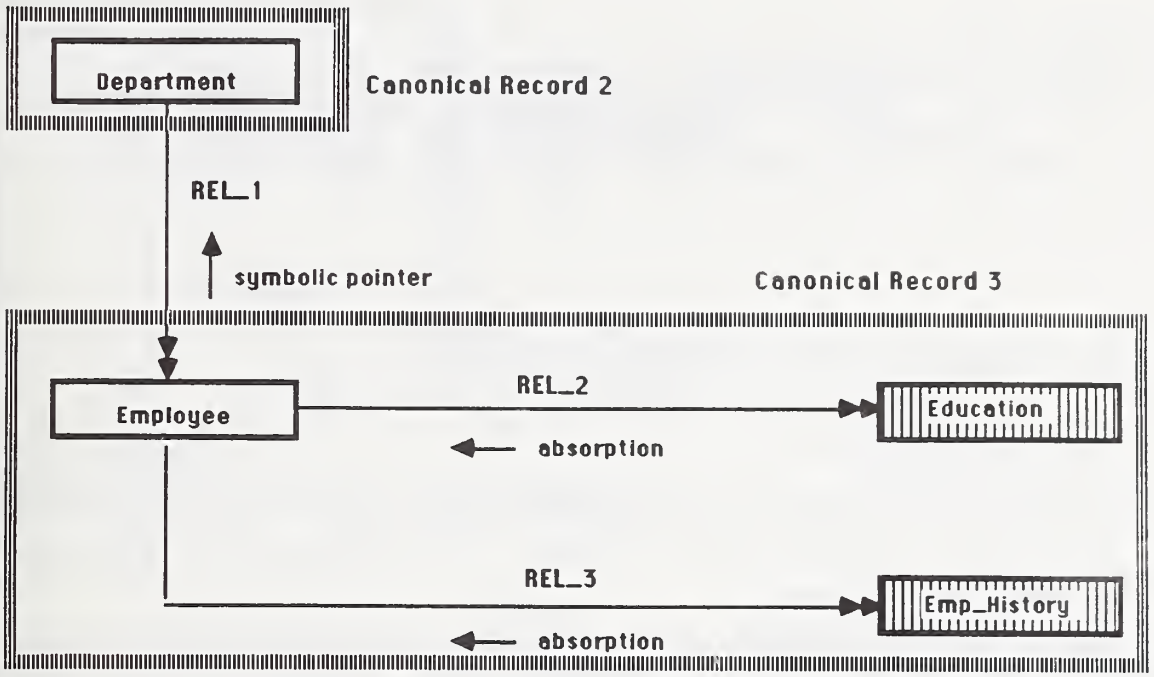


Itemized Costs for Entities and Relationships

Cost (DEPARTMENT)	=	386
Cost (EMPLOYEE)	=	5790
Cost (EDUCATION)	=	0
Cost (EMP_HISTORY)	=	0
Cost (REL_1)	=	6715
Cost (REL_2)	=	8394
Cost (REL_3)	=	0
		21285

The cost of accessing an entity is the sum of the time required to process updates and to directly access the entity in all the retrievals in which it is involved. The cost of a relationship is the sum of the time to process updates and to traverse that relationship in all the retrievals in which it is involved.

Figure 19. Skeleton 1



Itemized Costs for Entities and Relationships

Cost (DEPARTMENT)	=	7
Cost (EMPLOYEE)	=	5911
Cost (EDUCATION)	=	0
Cost (EMP_HISTORY)	=	0
Cost (REL_1)	=	44379
Cost (REL_2)	=	1361
Cost (REL_3)	=	0
		51658

Figure 20. Skeleton 2

Both skeletons have significant costs associated with accessing EMPLOYEE. In Skeleton 2 (Figure 20), REL_1 is represented with a symbolic pointer from EMPLOYEE to DEPARTMENT and has a retrieval frequency of 300,000 (see Retrieval 3 in Figure 21). REL_1 therefore has a high access cost associated with it in Skeleton 2, which is consequently less efficient than Skeleton 1.

```
{Retrieval 3}
SELECT  DEPARTMENT_NAME, EMPLOYEE_NAME, SSN, AGE
    FROM    DEPARTMENT, -- {FREQUENCY 500, PROPORTION 0.5}
        EMPLOYEE    {FREQUENCY 300000, PROPORTION 0.3}
    WHERE   EMPLOYEE.DEPT = DEPARTMENT_NAME AND AGE >50
```

Figure 21. Retrieval 3 from Figure 2

The following sequence of actions takes place to generate a better skeleton.

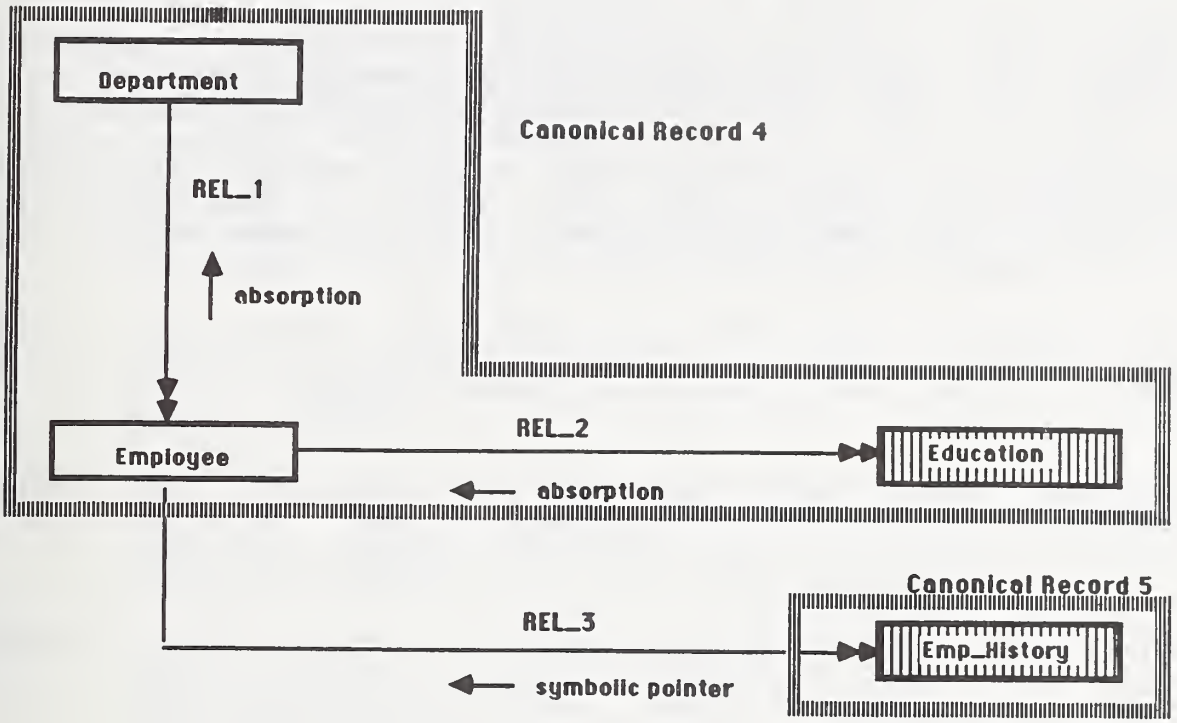
STEP 1. First, Skeleton 1 is selected for processing based on its lower cost.

STEP 2. At this point, the High Level directs that a relationship representation must be chosen for alteration. Rules for selection of relationships to be altered are applied to the problem. In Skeleton 2, the representation for REL_1 was changed to a symbolic pointer from EMPLOYEE to DEPARTMENT resulting in a high access cost. Since the representation for REL_1 has already been unsuccessfully varied it is not considered for alteration again. The high cost of accessing EMPLOYEE in Skeleton 1 triggers a rule which recommends another relationship along which to decompose the large canonical record. REL_3 is recommended because this relationship has less activity than REL_2.

STEP 3. REL_3 is selected for alteration.

STEP 4. The subsequent alteration of REL_3 results in Skeleton 3 shown below in Figure 22. Two new canonical records are created, Canonical Record 4 and Canonical Record 5. The substitution of a symbolic pointer from EMP_HISTORY to EMPLOYEE lowers the cost to 1723.

Skeleton 3 is now added to the list of skeletons for this cluster and may itself become the subject for further design activity. At the conclusion of processing the more efficient canonical record in Skeleton 3 would be chosen for fine-tuning using a conventional algorithmic design system over those of Skeletons 1 and 2.



Itemized Costs for Entities and Relationships

Cost (DEPARTMENT)	=	108
Cost (EMPLOYEE)	=	1614
Cost (EDUCATION)	=	0
Cost (EMP_HISTORY)	=	0
Cost (REL_1)	=	0
Cost (REL_2)	=	0
Cost (REL_3)	=	1
		1723

Figure 22. Skeleton 3

ACKNOWLEDGMENTS

We wish to acknowledge the contributions of John Carlis and Sal March of the University of Minnesota for their many years of research upon which this work is based and for their consultation without which this work would not have been possible; Thomasin Kirkendall for her advice and contributions on access methods; Mary Mitchell, John Cugini and Joseph Collica for their work in reviewing earlier versions of this report; Margaret Law for her contributions in refining the final version of this report; and to Leonard Gallagher for his work as a reviewer and for his valuable suggestions about SQL.

REFERENCES

[ANSI86] American National Standards Institute, Inc., American National Standard, Database Language SQL, ANSI X3.135-1986, New York, New York.

[CARL80] Carlis, John V., "An Investigation into the Modeling and Design of Large, Logically Complex, Multi-user Databases," Ph. D. thesis submitted to University of Minnesota, Minneapolis, Minnesota 55455, December 1980.

[DDLC78] Data Definition Language Committee, "CODASYL Data Definition Language Committee," Journal of Development 1978, Canadian Government Publishing Centre, Ottawa, Ontario, K1A0S9.

[FONG85] Fong, Elizabeth N., Henderson, Margaret W., Jefferson, David K., and Sullivan, Joan M., Guide on Logical Database Design, NBS Special Publication 500-122, National Bureau of Standards, February, 1985.

[MARC78] March, Salvatore T., Jr., "Models of Storage Structures and the Design of Database Records Based Upon a User Characterization," Ph.D. thesis submitted to Cornell University, May 1978.

[SHAF76] Shafer, G., A Mathematical Theory of Evidence, Princeton University Press, Princeton, 1976.

[THOM85] Thompson, Terence R., "Parallel Formulation of Evidential Reasoning Theories," Proceedings of the Ninth International Joint Conference on Artificial Intelligence, Los Angeles, CA, 1985.

NBS-114A (REV. 2-8C)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO. NBS/SP-500/151	2. Performing Organ. Report No.	3. Publication Date February 1988

4. TITLE AND SUBTITLE

Computer Science and Technology
A Knowledge-Based System for Physical Database Design

5. AUTHOR(S)

C.E. Dabrowski and D.K. Jefferson

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL BUREAU OF STANDARDS
U.S. DEPARTMENT OF COMMERCE
GAITHERSBURG, MD 20899

7. Contract/Grant No.

8. Type of Report & Period Covered

Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

Same as Item 6.

10. SUPPLEMENTARY NOTES

Library of Congress Catalog Card Number 88-600502

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

A knowledge-based system for physical database design has been developed at the Institute for Computer Sciences and Technology. This system processes large multi-entity databases with complex workload requirements and identifies near-optimal physical designs. It employs heuristics developed by physical design experts and cost modeling algorithms to reduce the large number of design alternatives available in large complex problems to a few select designs. This system is implemented in Lisp.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

certainty factor; entity-relationship model; inference engine; knowledge engineering; knowledge-based system; logical data structure (LDS); logical database design; physical database design; rule-based system

13. AVAILABILITY

☒ Unlimited

☐ For Official Distribution. Do Not Release to NTIS

☒ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.

☐ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

59

15. Price

USCOMM-DC 6043-P80

ANNOUNCEMENT OF NEW PUBLICATIONS ON COMPUTER SCIENCE & TECHNOLOGY

Superintendent of Documents,
Government Printing Office,
Washington, DC 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Bureau of Standards Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

(Notification key N-503)

☆U.S. Government Printing Office : 1988 -201-597/82529

NBS Technical Publications

Periodical

Journal of Research—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NBS under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NBS publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NBS publications—FIPS and NBSIR's—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Bureau of Standards
Gaithersburg, MD 20899